MIT/LCS/TR-266

AN EASE OF USE EVALUATION OF AN INTEGRATED EDITOR AND FORMATTER

Michael Good

An Ease of Use Evaluation of an Integrated Editor and Formatter

Michael Good

November 1981

Massachusetts Institute of Technology
Laboratory for Computer Science
Cambridge Massachusetts 02139

This empty page was substituted for a blank page in the original document.

An Ease of Use Evaluation of an Integrated Editor and Formatter

by

Michael Good

An earlier version of this report was submitted to the
Department of Electrical Engineering and Computer Science on
August 21, 1981 in partial fulfillment of the requirements
for the Degree of Master of Science in Computer Science

Abstract

Etude is an integrated text editor and formatter that was designed to be easy to learn and easy to use. To measure Etude's success in meeting these goals, twenty-one computer-naive temporary office workers were taught to use Etude in a controlled experiment. Ninety percent of the subjects were able to create and edit letters after a training period of less than two hours and twenty minutes, though they were not able to perform these tasks as quickly as they could when using a typewriter. Etude did not appear to have any systematic effect on subject anxiety. The subjects had favorable attitudes towards using Etude. These attitudes were as least as favorable as their attitudes towards using a typewriter.

Thesis Supervisor: Michael Hammer
Title: Associate Professor of Computer Science

Key words: Human factors
Man computer interface
Office automation
Text processing

Table of Contents

Table of Figures

Table of Tables

Chapter One

Introduction

Designers of systems that are intended to be easy to use have many guidelines available to them in the literature. Most of these guidelines are based on the intuition and experiences of particular designers with particular systems. Very few of them have been evaluated experimentally, so one must be cautious not to attribute more authority to these guidelines than they deserve.

If a computer system follows these guidelines, is the resulting system easy to use? This question cannot be answered in general, but it can be asked of particular systems that have been designed from the outset with the intention of being easy to use. Experimental evaluations of such systems can contribute to an understanding of the usefulness of these guidelines, as well as providing a way to measure the success of the system in meeting the goal of ease of use.

When interpreting the results of an experiment, the issues of internal and external validity [74] must be addressed. An experiment has internal validity if the experimental method being used accurately addresses the questions which the experiment is intended to answer. Extraneous factors must be eliminated or controlled. An experiment has external validity if the results can be generalized beyond the specific questions addressed in the experiment.

Thus, an ease of use evaluation of a computer system has internal validity if the experiment actually does measure the aspects of ease of use which the designers claim for the system. If the techniques used to make this system easy to use are shown to be extendable to other systems, the experiment has external validity as well.

To ensure internal validity and increase the likelihood for external validity, an ease of use evaluation must address several questions:

- What system is being evaluated? Many descriptions of computer systems do not clearly differentiate between the system envisioned in the specifications and the system actually implemented. This is not necessarily a bad thing when the goal of the description is to present the ideas behind the system, but it is a major flaw if the system is being evaluated experimentally. Details that would hinder a general presentation become very important when a system is actually being used.

- How does the system's design reflect the goal of being easy to use? If the specific design guidelines that were used are not carefully presented, a subsequent evaluation reveals little about the utility of the guidelines. An analytical evaluation of a system's adherence to guidelines, completed before the experiment is actually run, is the only resource available to give this type of evaluation any amount of external validity.

- What aspects of ease of use are being claimed for this system? The specific criteria that are being used to determine ease of use must be carefully stated. Otherwise, it is very difficult to determine if the hypotheses used in the experiment are valid tests of the designers' claims, and the experiment's internal validity is questionable.

- How is the system being evaluated? An experiment's internal validity rests on the soundness of its design and the propriety of its administration.

This report presents an ease of use evaluation of Etude, an interactive and integrated text editor and formatter developed by the Office Automation Group of the MIT Laboratory for Computer Science. As the user creates, edits, and formats a document, Etude displays the results on its full page, high resolution bit-map display screen. Etude is the first component of an integrated office workstation that will include functions such as business graphics, database/file management, electronic mail, and a calendar. All of these functions will be integrated into a single system with a uniform and consistent interface. The user will not have to switch back and forth among various systems to accomplish a certain task.

One of the workstation's principal design goals is that it be easy to use. Users of these workstations will not necessarily know anything about computers, but they must be able to learn the system quickly and use it efficiently in order for it to be accepted. Etude was designed to be both easy to learn and easy to use. A prototype version was evaluated according to the guidelines for building easy to use systems that are currently available in the literature. In most cases, Etude measured up to the guidelines. Changes were made where this was not the case.

Ease of use is a multi-faceted problem. Many guidelines are contradictory, for increasing one aspect of ease of use may in turn decrease another aspect. Many current systems exhibit a tradeoff between ease of learning and ease of use once the system is learned. Those that are easy to learn are often hard to use due to redundant or verbose features of the interface, while systems that have a very terse interaction style may be easy for experts to use but very difficult for novices to learn.

A system that follows ease of use guidelines may turn out to be easy to use in one aspect but not in another. The goals of each system determine the relative importance of the various factors of ease of use. In the case of Etude, the following four general criteria were chosen to represent the notion of ease of use:

1. Ease of learning. Etude should be easy for a completely computer-naive person to learn. Such a person should be able to use Etude for useful work after a short, informal training period.

2. Ease of use once learned. Etude should be easy for people to use. Users should be able to create and edit documents quickly, without being burdened by a clumsy or slow interaction style.

3. Anxiety factor. Etude should not induce anxiety in its users. Common anxieties in computer users include the fear of breaking something and the fear of losing a large amount of work without notice.

4. User attitudes. Both novices and experts should enjoy using Etude.

An experiment was designed to evaluate Etude according to these gt.neral criteria. This process involved the refinement of these criteria into hypotheses suitable for an experimental evaluation. Twenty-one computer-naive temporary office workers served as subjects. The experiment revealed that while the current version of Etude succeeds in meeting the criteria of ease of learning and user attitudes (and perhaps the anxiety factor as well), it does not meet the criterion of ease of use once learned. The slow response time of the current version of Etude may be the major factor in not meeting this last criterion. This provides some evidence for the usefulness of the current body of user interface guidelines, though the experiment by itself has little demonstrable external validity.

The questions raised in this chapter are answered in detail in the remainder of this report. Chapter 2 describes the version of Etude that was evaluated in this study. Etude's adherence to user interface design guidelines is analyzed throughout this description. Chapter 3 refines the four general criteria mentioned above into the more specific criteria used in the evaluation. Chapter 4 discusses the design and administration of the experiment. The results are presented and discussed in Chapter 5. Chapter 6 examines questions that were not dealt with in this study that are topics for future research.

Chapter Two

Etude

Etude has been implemented three times since it was first designed in 1979.

1. The *first version* of Etude was finished in Spring 1980. A demonstration of this version was the basis for an article in the *Seybold Report on Word Processing* [23]. Ilson's Master's thesis [48] includes a description of the original design process as well as details of the first implementation. A published paper [39] also describes the implementation of this version.

2. The *experimental version* of Etude is the one that was actually evaluated in this study. It is based on the first version, but includes several changes that I made as a result of my analysis of the first version's adherence to user interface design guidelines.

3. The *new version* of Etude is currently being implemented by members of the Office Automation Group. Many changes have been made in this version. The software architecture has been completely revised to allow for integration of the other workstation tools that are being developed. Functions that were omitted from the prototype are now being included, and details of the user interface have changed. The published overview of Etude [40], the current specifications [49], and my paper discussing Etude's adherence to user interface guidelines [37] all refer to the new version of Etude.

This report was originally intended to evaluate the new version of Etude. However, when it became clear that this version would not be ready in time for the scheduled evaluation, the experimental version was created to allow the evaluation to go forward.

This chapter describes Etude's evolution, emphasizing how the experimental version of Etude meets user interface guidelines available in the literature. Etude's design goals are briefly discussed and some general concepts are explained. This is followed by an overview of the first version of Etude, drawn from an earlier internal memo [36]. The differences between the first and experimental versions are then discussed in detail. Most of the remaining problems with the experimental version were anticipated in the design of the new version of Etude; a discussion of these problems completes the chapter.

The descriptions of Etude in this chapter assume a familiarity with text processing systems. The tutorial in Appendix A, starting on p. 105, is a basic introduction to the experimental version of Etude.

2.1 Design Goals

While Etude may be used from conventional CRT terminals, it is intended to be used on a powerful stand-alone computer system with a full page high resolution bit-map display.[1] This gives Etude extensive formatting capabilities, such as displaying proportionally spaced text with multiple fonts and type sizes. Etude can show the user how his document would look if typeset while he is working with the document.

The primary design goal for Etude was to develop a system with these capabilities that was both easy to learn and easy to use. People with no training in either the use of computers or typography should be able to sit down in front of the system and learn how to produce a formatted document. In order for this to happen, some kind of online assistance facility must be provided.

Many computer systems that are easy to learn also turn out to be awkward to use once learned. Indeed, the goals of ease of use and ease of learning often are contradictory.

[1] The experimental version runs on a large time-sharing computer which is connected to a smaller computer that has a bit-map display. This issue will be discussed several times in this chapter.

This is especially true when the simplicity desired for ease of learning conflicts with the flexibility desired for ease of use [32]. One of Etude's challenging goals was the specification that the facilities that are included to aid the novice should not encumber the expert.

Another goal was to reduce the anxiety factor that is commonly associated with using computer systems. This has been likened to the feeling of "walking a tightrope while wearing a blindfold" [78]. Many user interfaces are obscure enough that the user cannot be sure of the result of a given operation. Even when the interface is relatively straightforward, the user is usually unable to reverse the results of an operation if they are not what was intended. In other words, the user may not know if his next step will lead him off the tightrope, but he does know that once off the tightrope, he won't get back on in one piece.

The last design goal was that Etude should automatically format a document in real time without requiring the user to directly specify the typographic details of a document's appearance. Very few of Etude's users would have any experience in typography. Instead, Etude's interface should be on a level that is natural to the user and his application.

These goals led directly to some of the major characteristics of Etude's user interface. Ease of learning is aided by Etude's simple command structure. Etude's commands are structured like commands in English, in that they are composed of a verb and one or more objects. Most objects contain a noun and optional modifiers. Online assistance is available at any time, and a tutorial for novices is included.

In order to make Etude easy to use, the most common vocabulary items can be specified by pressing a special key marked with the full name of the item. Items that are not available on special keys may be selected from a menu or typed in full by a new user; a more experienced user can type in an abbreviated form. The expert user can create his own abbreviations in addition to those provided by Etude.

To reduce the anxiety factor caused by the fear of irreversible error, Etude provides an undo facility. This allows the user to reverse the effects of an arbitrary sequence of operations, so that he can get back to a particular spot in his interaction before the error was made. The undo facility also encourages experimentation, which aids the learning process.

Many of Etude's nouns refer to the natural structure of a document; for instance, a user typing a letter can refer directly to the return address, a specific paragraph, or other parts of the letter. This lets the user format a document in a more high-level fashion than by referring directly to margins, spacing, type sizes, or other lower level typographic features.

2.2 General Concepts

Certain general concepts have remained present throughout the evolution of Etude, though the terminology has changed in several instances. The discussion of these concepts is based on a similar discussion in the new Etude specifications [49], but emphasizes how these concepts correspond to available guidelines for user interface design available in the literature.

The first version of Etude was designed by incorporating what were considered to be the best features of several existing editors and formatters [48]. The analysis with respect to user interface guidelines was done as part of the review of the first version, and resulted in several changes in the experimental version. Some of this analysis has been previously published in more detailed form [37].

2.2.1 Document Structure

Etude deals with three aspects of a document's structure—the *content*, the *editorial structure*, and the *outward appearance.* All text editing systems work with the content of a document. They also deal with the outward appearance of a document, which is the way that the content appears on either a display screen or a printed page. This includes the way in which a document is broken into pages, columns, and lines; the size of the margins and the spacing between lines; and the type styles and sizes that are used.

In the simplest editing systems, the outward appearance of a document depends solely on the content of a document. Line breaks are specified manually. Page breaks are done automatically by the output device (usually a line printer or a hard copy terminal), but can be modified by inserting blank lines or page feeds into the content of a document. In most formatting systems, the outward appearance is controlled by using formatting commands to specify changes in margins, spacing, type style, etc.

Some recent systems link the outward appearance of a document to both its content and its editorial structure. The editorial structure is the classification and organization of the information and ideas contained in a document. For example, a business letter usually contains a return address, an address, a greeting, a body with paragraphs, a closing, and some notations. These are some *components* of its editorial structure. In Etude, the user is encouraged to identify these components and refer to them as he types and edits the document. The Scribe formatter [86] was the model for Etude's idea of editorial structure. A similar idea is also present in the Generalized Markup Language [35]. The use of editorial structure lets the user deal with familiar concepts rather than arcane details. Newman and Sproull [75, p. 448] point out that this lets the user model the system in a more natural way.

2.2.2 Objects, Regions, and Cursors

The Etude user works with many different types of *objects.* If he is editing a letter, the letter is an object he is working with. The components of a letter's editorial structure are objects contained within the letter object. There are also built-in objects such as characters, words, lines, sentences, columns, and pages.

Etude operations generally involve objects, collections of objects, or positions between objects. A user can erase a paragraph (an object) or move three words (a collection of objects) to the middle of another sentence (between objects). A *region* contains an object or collection of objects; a *cursor* indicates a position between objects.

2.2.3 The Display

Etude's display screen is divided into several *windows.* The main *text window* contains the content of the document, with a cursor positioned where the user is working. There is a long, thin *format window* to the left of the text window. The format window appears as if it were in the document's left hand margin, and contains the names of the components of the document's editorial structure.

At the top of the screen is the *interaction window,* which is used for communication of information between Etude and the user. It contains a command line, a response line, and an environment line. The command line echoes command keys as they are struck, inserting prepositions, plural forms, and other function words[2] as necessary. It is also used to display prompts. The response line is used to display system messages. The environment line contains miscellaneous information about aspects of the text surrounding the cursor.

[2] Function words also include articles, conjunctions, pronouns, and auxiliary verbs [100].

This display layout incorporates several ideas present in ease of use guidelines. Feedback is provided for every keystroke, either through a change in the text window or through the echoing process in the command line of the interaction window. This follows Gaines and Facey's guideline of "immediate feedback" [30], which they viewed primarily as a way of preventing errors. However, systems with such immediate feedback are becoming more popular as evidence of their power and attraction accumulates. This can be seen in areas ranging from computer games [62] to integrated editors and formatters and other "what you see is what you get" systems, such as VisiCalc [114].

Other guidelines are reflected in the interaction window. For example, Etude displays an "[ok]" in the command line when it starts to work on a command, which provides faster feedback for commands that take a long time to complete; this idea has been promoted by many authors as a way of reducing user frustration [75, 77, 94]. The response line provides a consistent place for displaying error messages, as recommended by Rohlfs [92]. Also, the general idea of an interaction window provides at least some limited support for what Thomas [112] calls a "metacomment" facility, in which the system and the user exchange information about the interaction itself.

2.2.4 Command Structure

Etude's commands are similar to commands in English; they start with a verb and contain one or more objects. Some verbs take a direct object (often a region), and some also take an indirect object (usually involving the movement of a cursor).

Many Etude commands follow verb-modifier-noun form. Verbs include commands such as **go to**, **erase**, and **move**.[3] Modifiers include **next**, **previous**, **start of**, **end of**, and

[3] Throughout this report, Etude vocabulary items are represented using **boldface**.

positive integers. Nouns include built-in objects like **word** and **sentence** as well as components such as **paragraph and bold**. All of the modifiers and built-in objects are assigned to individual keys on the Etude keyboard, as are the most common verbs and component names.

Commands are formed by combinations such as **go to previous line, erase 3 words,** and **move sentence (to) end of next paragraph.** Words shown in parentheses and the "s" to form plurals are provided by Etude when echoing the command in the command line. This prompting is similar to the use of "noise words" in TENEX [7].

This command structure has several advantages:

- An experiment by Ledgard et al. [56] showed that subjects performed better with a line-oriented text editor that used English phrases as commands than they did with an functionally identical editor with a more traditional command structure. These improvements held over three different measurements of performance and over three different levels of user experience.

- Bennett [3] claims that verb-object form is easy to teach and can serve as a memory aid. This is related to Treu's theory [113] that verb-object form results in less mental work for the user.

- Combining verbs and objects to form commands is efficient as well. As Watson [119] points out, a set of m verbs and n nouns gives $m \times n$ commands but uses only $m + n$ vocabulary items. This efficiency is even greater when modifiers are included.

One problem associated with both natural language and less ambitious "English-like" systems is the potential for fooling the user into believing that the system understands more than it really does. This can lead to errors when users try to specify an English-like command that is beyond the capabilities of the system, as Plum [83] and Palme [81] have pointed out. Boden [8] has worried about the dehumanizing potential of computers that appear to be more intelligent than they really are. Natural language systems also are prone to problems with ambiguity, as Hill [47] points out. Since Etude falls in the class of "English-like" systems, the ambiguity problem is not a major concern. Fooling the user is a possible problem that will be discussed in more depth in the more detailed sections below.

2.2.5 Pointing

Many operations require that the user point to a particular position in the text. This corresponds to moving the main cursor to a new position in the document. A simple way to do this is to use a *pointer.* One example of a pointer is the set of cursor positioning arrows (or "step keys") on the Etude keyboard which move the cursor up, down, right, or left.

Although this is the type of pointer that has been used in all the versions of Etude implemented to date, there is nothing that restricts the choice of physical device that can be used as a pointer. A device such as a "mouse" or a joystick might be incorporated later. An experiment by Card, English, and Burr [13] suggests that a mouse is superior to other pointing devices, but the devices tested included a slow set of step keys (15 cps in the horizontal direction) and only one particular type of joystick.

2.2.6 User Aids

The Etude user may press the **help** key at any time. Etude will respond by displaying some information indicating what the user is currently doing and what his options are. If the user is not involved in the middle of a command, Etude will also indicate the last few operations that the user performed.

The idea of a help facility or a more general form of online assistance is well established [65, 81, 89, 118]. Besides being of use to novices, it can help refresh the memory of an infrequent or discretionary user [4]. Information about what operations have been performed recently can be helpful for a user trying to remember his place in the dialogue [25, 43, 57]. This is useful not only when the user is confused, but also when the user returns to the system after an interruption.

The concept of menu selection is also important to Etude. Theoretically, the user may press **menu** at any time and see a list of options displayed on the screen, with the default option highlighted by being displayed in reverse video. By using a pointer, the user can move the current selection from the default option to the item that he wants; the current selection is always highlighted. The confirmation key may be pressed to indicate that the selection has been made, or the **cancel** key may be pressed to get rid of the menu. The user may also type in the name of the item after the menu is displayed, instead of selecting it with the pointer.

An experiment by Fields, Malsano, and Marshall [26] compared four methods for inputting tactical data. These methods included typing of code names, typing of code names with spelling correction, menu selection of English names, and typing of either code names or English names with automatic completion provided. Menu selection was the most accurate of the techniques and was not slower than the others. The experimenters took care to point out that the subjects in this study did not work with the system long enough to become expert users. They also criticized the use of a trackball in menu selection.

In both the first and experimental versions, the use of menus was largely limited to the selection of component names. In the new version of Etude, the **help** and **menu** keys will operate in a more extended and integrated fashion, as described in section 2.4.9.

The use of a confirmation key and a cancellation key described with menus extends to many other commands. Any command that makes a substantial change to a document (such as **move**, **copy**, and **erase** when applied to large regions) must be confirmed before it is carried out. The regions involved are highlighted before confirmation is requested. Pressing **cancel** aborts the command, returning the user to the state he was at before starting to specify the command.

The idea of confirming commands that may have serious consequences has been expressed by Engel and Granda [25] and others. The process of highlighting and confirmation, applied to the **erase** command, is similar to Rohlfs' idea of an erasing function that simulates pencil and paper, where erasure still leaves a faint trace of the original before it is written over. The **cancel** key embodies ideas about a reset key proposed both by Gaines and Facey and Gilb and Weinberg [34].

If an operation has been completed but the results are not what was intended, the user may press **undo** to reverse the effects of that operation. In theory, this can apply to a sequence of operations as well, but user interface restrictions limited the effect to a single operation in the first and experimental versions.

Again, the idea for an undo function has been around for some time. Advocates have appealed to its error correcting capability as both a desirable feature by itself [19, 30, 42] and as a way of relieving anxiety [4, 34] and user frustration [27]. The undo key also encourages experimentation and a learning by doing approach, which Jones [51] believes to be more important to "natural" communication than having a "natural language" interface.

2.3 The First Version

2.3.1 Noun Phrases

In Etude, noun phrases can be used to describe regions of text and locations in text. A noun phrase contains one noun, optionally preceded by one or more modifiers.

Etude's modifiers are **start of**, **end of**, **next**, **previous**, and the positive integers. The first version of Etude places some constraints on the way that modifiers can be combined. If an integer is present, it must be the last modifier in the phrase. The modifiers **start of** and **end of** are mutually exclusive, as are the modifiers **next** and **previous**. A modifier from the former pair cannot follow a modifier from the latter pair. Thus **end of next 10 sentences** is a valid noun phrase, but **next previous word** and **next start of paragraph** are not allowed.

The restriction that noun phrases must end with a noun means that phrases such as **page** 3 are not allowed. This is a limitation in the first version that will be removed in the new version of Etude.

Etude's nouns include built-in objects, component names, search strings, and label names. The built-in objects are **character**, **word**, **sentence**, **line**, **paragraph**, **column**, **page**, and **document**. The component names available in letters and reports are listed in figures 2-1 and 2-2, respectively.

A search string is an arbitrary string of characters, preceded either by a single quote or a double quote character. The end of the search string is indicated by pressing the confirmation key, which may be preceded by a matching quote character. If the user includes spaces in his search string, the search operates on words, ignoring the difference between different types of spacing between words. Miller and Thomas [67] recommend this as an improvement over many current search facilities.

paragraph
bold
italic
return address
address
greeting
body
closing
notations
ps
postscript
cc
xc
item
list
number
outline
center
flushright
flushleft
nojust
doublespace
narrow
letter
insert

Figure 2-1: Components Available in a Letter

paragraph
bold
italic
chapter
section
subsection
chaptertitle
sectiontitle
subsectiontitle
majorheading
heading
subheading
item
list
number
outline
description
outdent
format
verbatim
quotation
center
flushright
flushleft
nojust
doublespace
narrow
report
insert
hdx

Figure 2-2: Components Available in a Report

Label names are names given by the user to regions or cursor locations. The **label** command, described below, is used to assign these names.

2.3.2 User Aids

The **help** and **menu** keys work as described above. The name of the confirmation key is **execute**; it is used to confirm dangerous commands, to indicate selection from a menu, and to indicate the end of typed names, such as those of components or labels.

If **execute** is struck before the name is completely typed, Etude will attempt to complete the name. If there is only one available name that starts with the given string, the completed name is echoed in the command line. If there are no names that match the given string, Etude displays a message in the response line telling the user that no names matched and displays the string up until the point where it could not find a matching character. If more than one name matches the string, Etude displays the string up to the point where it becomes ambiguous. It also displays a message telling the user that the string is ambiguous.

Suppose a user wanted to specify a "flushright" component. If he just typed "f" and pressed **execute,** Etude would display "flush" and inform the user that the string is ambiguous (since there is also a "flushleft" component). If the user then hit an "e" instead of the "r" and then pressed **execute,** Etude would once again display "flush" and inform the user that none of the names matched. If the user then typed "r" and pressed **execute**, the system would display the completed name "flushright" in the response line.

If this scheme sounds complicated to the reader, he will not be surprised to learn that the experiment by Fields, Malsano, and Marshall mentioned above found that an identical completion scheme was the most error-prone of four different inputting methods. Ledgard, Singer, and Whiteside contend that abbreviation facilities are often the least well designed part of an interactive system (p. 16). However, Nickerson and Pew [77] point out that automatic completion combines goals from both ease of use and ease of learning. Changes were made to this scheme in both the experimental and new versions of Etude, as will be discussed below.

Another user aid is the **again** key, which repeats the last command when possible. As many commands would be tricky or dangerous to repeat automatically, **again** works primarily with simple commands such as **go to** and **delete.**

2.3.3 Verbs

Etude's verbs can be divided into four categories:

1. Cursor movement (↑, →, ↓, ←, **go to**)

2. Region definition (**begin, end**)

3. Editing (**erase character, erase word, delete, move, copy, label**)

4. Formatting (**begin format, end format, format, unformat, new format, merge previous, merge next, anchor**)

These verbs are described below, as are some auxiliary commands that are not part of the regular command structure.

Cursor movement can be done either by using a pointer (in this case, the arrow keys ↑, →, ↓, and ←) or by using the **go to** command followed by a noun phrase. If the noun phrase starts with **start of, end of, next**, or **previous**, the verb **go to** can be omitted.

As mentioned before, a noun phrase is often used to define a region. However, the **begin** and **end** keys can also be used to define arbitrary regions that would be difficult to specify using a noun phrase. After the operation requiring a region is initiated (e.g., by striking the **delete** key), pressing **begin** starts the region definition. Any sequence of cursor movement commands can now be used. Pressing the **end** key marks the end of the region.

The cursor position where the **begin** key was struck does not have to be before the position where the **end** key was struck. Also, the **begin** key may be struck again at any time before the **end** key is struck. This resets the starting point to the current cursor position.

For the remaining verbs, we will use <region definition>, <cursor movement>, and

<component name> to indicate that these items are used as arguments to the given command.

Erase character erases the character immediately to the left of the current cursor position. **Erase word** erases the word immediately to the left of the current cursor position; if the cursor is positioned in the middle of a word, the entire word is erased. No confirmation is required.

Delete <region definition> removes the defined region from the document. **Move** <region definition> (to) <cursor movement> moves the defined region to a new position in the text, as specified by the cursor movement. **Copy** differs from **move** only in that **copy** duplicates the region, whereas **move** deletes the region from its original position. All these verbs highlight the region after it has been defined and require confirmation.

Label allows for the labelling of a cursor or a region. After typing **label** the prompt "(cursor or region)" is displayed. Either "cursor" or "region" should be typed here, though **execute** can be used to provide completion. If a cursor is being labelled, the user may then move the cursor, pressing **execute** when finished. If a region is being defined, the user defines a region in the standard way; the region is then highlighted. After specifying what is to be labelled, the user types in the name of the label, ending the name by pressing **execute**. Any label name may be used as a noun when specifying cursor movement; the name of a labelled region may be used for a region definition as well.

Begin format <component name> starts a new component at the current cursor position. **End format** is used to mark the end of the smallest component which contains the current cursor (hereafter called the *current component*) when typing in text. It moves the cursor just past the end of the current component and does not require confirmation.

Format <region definition> (as) <component name> is used to add a component to an already existing region of text. Again, the region is highlighted after it is defined, and confirmation is required.

Several formatting verbs operate on the current component and thus do not take an argument. Besides **end format**, these verbs include **unformat**, **new format**, **merge previous**, and **merge next**. **Unformat** removes the current component from the document's editorial structure while leaving the content of the component in the document. **New format** splits the current component into two components of the same type at the current cursor position; the cursor moves to the beginning of the second component. **Merge previous** combines the current component with the previous component of the same type; it only works if the two components are adjacent to each other. **Merge next** works in the same way but in the other direction. Except for **new format** (and the aforementioned **end format**), these commands require confirmation.

Anchor <picture name> (to) <cursor movement> is a command that leaves blank space for a picture within a document. The only picture name available in the first version is "2X2Picture," which leaves room for a two inch square picture. Other shapes could have been provided using the same underlying mechanism. Space for the picture will be provided as close as possible to the position specified by the cursor movement. Confirmation is required. In this version, **Anchor** only works with the Paper document type.

Several other commands were provided in Etude that were not part of the regular command structure. These are commands that handle such functions as I/O and screen display. Several commands that are used only for debugging purposes are not included here, but are listed in a full description of the first version [36].

Etude files can be read from or written to disk by using the **read file** and **write file** commands. After the command is specified, the command line will prompt the user to specify whether the file is in **image** or **script** form. In nearly every case the file is in image form; script form was used during an early attempt to instrument Etude. After pressing **execute** to indicate the completion of the image/script response, the user types the name of the file to be read or written, followed by **execute.**

Three commands affect the display of the screen. **Redisplay** clears and redisplays the entire screen. **Update status line** redisplays the interaction window. The **Change flag** command can be used to turn display of the format window off and on. Flags may be set, reset, or toggled. **Change flag** also controls two other flags that are used only for debugging purposes. The selection of set, reset, or toggle and the flag to be changed are prompted in the response line.

Change subdocument lets the user move between the main body of text (the "bodytext" subdocument) and other parts of the document, such as headers and footers. The user types the name of the subdocument after specifying the command. The Paper document type is the only one which can make use of this command in the first version.

2.3.4 Hardware and Implementation Issues

While Etude is intended to run on a stand-alone computer, the first version was implemented on a large time-sharing DECSYSTEM-20, using a Nu terminal [117] as a display device [76]. The Nu terminal includes a bit-map display of 800 × 1024 pixels. The program could also be run on a traditional terminal such as a DEC VT100 or a Zenith H19. The first version was written in the CLU programming language developed at MIT [60].

The first version was also limited to using an old prototype version of the Nu, based on an Intel 8086 microprocessor. This version of the Nu had only a standard computer terminal keyboard, without any of the special keys envisioned in the design of Etude. This meant that the special vocabulary items were invoked by using one key as a control or "code" key, which was held down together with another key to produce the desired item.

2.4 Improvements

Several changes had to be made before the first version of Etude could be made suitable for an evaluation. These changes included:

- Changing parts of Etude's vocabulary in accordance with user interface design principles.

- Improving the handling of component names.

- Changing the wording of system messages, including those provided by **help** and by error messages.

- Changing Etude to use the new version of the Nu terminal, based on a Motorola 68000 microprocessor. This version of the Nu included a large programmable keyboard. Along with this, the actual workstation setup had to be devised.

- Changing the display layout to eliminate several problems. For example, the font used by Etude in the first version was chosen for its usefulness in giving demonstrations to large numbers of users. It was too large to fit an entire page onto the supposedly full page display.

- Writing a tutorial for Etude.

- Making Etude more reliable, both by debugging the program and by providing an automatic backup facility.

- Instrumenting Etude so that usage patterns and problems could be studied.

The approach taken to these problems is detailed in this section. Many of these problems involve "minor" details, but they are especially important because they effect nearly every element of a user's interaction with the system [57].

2.4.1 Vocabulary Items

Several of Etude's vocabulary items were changed in the experimental version. These changes were simple substitutions of one word or phrase for another.

Although the original choices of vocabulary tried to avoid "computerese," some poor choices of words still crept in. The worst offender was the term "hlto," which was changed to "component." The problem of what to call this object is not a simple one. Although components usually contain format information, this is really incidental to the semantic information that is represented. Components identify the editorial structure of the document, which includes more than specifications of the outward appearance. Thus the common term "format" is not appropriate for a component. We could not think of a shorter name that was completely appropriate and avoided inaccurate or undesirable connotations. Although "component" is not a familiar word to most people, it is far better than "hlto," a term derived from an out-of-date acronym that does not follow English word formation rules.

The formatting commands also needed an overhaul. As all of them contained the word "format" in their names, they were not completely appropriate for the reasons discussed above. They were also not easily generalized across the other tools that would be integrated with Etude. Thus several changes were made. **Format** and **unformat** were replaced by **make** and **remove**. Combined with the unimplemented **change** command, these new commands should be useful in other areas of the workstation's operation, such as database management. The **begin format** and **end format** commands were eliminated; their functions were taken over by **begin** and **end**. The useful **new format** key was simply renamed **new component**. In addition, a general **component** object was added to the set of built-in objects.

Other poor choices of vocabulary include **execute**, with its connotations of firing squads; **merge**, a word seldom found outside of highway signs; and **delete**, a remnant of computer terminology. **Go ahead** replaced **execute**; **combine previous** and **combine next** replaced **merge previous** and **merge next**; and **erase** replaced **delete**. Also, **back space** and **back word** replaced **erase character** and **erase word**.

One tricky area involved the I/O commands. The terms **read file** and **write file**, as well as similarly worded commands found in a great many systems, can be confusing to naive users who understand neither what is being read or written nor who is doing the reading and writing. The words "read" and "write" simply mean different things in computer terminology than they do in common English usage [93].

To avoid this problem, an analogy was drawn to a filing system. The computer operation **read file** was replaced by the command **retrieve document**. This operation involves retrieving a document from Etude's filing system and displaying it on the screen. Similarly, **write file** was replaced by **file document**, which files the document away for future use. Section 11 of the tutorial (Appendix A) explains these operations to the new user. Since script form was never used, the "file/script" distinction and prompt were eliminated from the command. As in the first version, there is no operation to produce a hard copy version of the document.

2.4.2 Component Names

In the first version of Etude, the **paragraph** key could not be used in some situations where the user could type in "paragraph" as a component name. This problem was fixed in the experimental version, which also added the component names **bold** and **italic** to the set of special keys.

The scheme for completing component names was also modified slightly. If more than one component started with a given string, the experimental version of Etude chooses a default from one of these names, instead of reporting the ambiguity as it did in the first version. This change permits the user to specify common component names with single letter abbreviations. In a letter, for example, **return address** can be abbreviated to "r," **address** to "a," and so on for **greeting**, **paragraph**, **closing**, and **notations**. While not ideal, the new scheme can avoid some of the cumbersome problems of the earlier version.

The default name is the component name listed earliest in the data base. Figures 2-1 and 2-2, shown previously, list the components in the same order as they appear in the database, with the components in the first column preceding all those in the second column. For example, the default for "i" is **italic**, not **item**.

2.4.3 System Messages

The problem of system messages has been addressed by Shneiderman [99], who worries about the preponderance of violent (e.g., "fatal error," "catastrophic," "disaster") and obscure (e.g., "syntax error," "OC7, OC4," "?") terms in error messages and the lack of information that would help the user correct his error. He has conducted studies which show that replacing error messages like the "?" in the UNIX[4] text editor *ed* with short but more specific error messages improves user performance and satisfaction. The effect of the tone of the messages (hostile, neutral, or courteous) was less clear cut, but appeared to indicate that user satisfaction was increased by having courteous messages. This provides evidence to back the guidelines that error messages be polite and specific, avoiding terminology unknown to the user. These guidelines have been proposed by many authors [5, 19, 53, 82].

[4] UNIX is a registered Trademark of Bell Laboratories.

This goal is not completely satisfied in the experimental version of Etude. In general, the messages are not specific and direct enough; a general purpose message is used where a more specific message would be preferable. However, some of the worst messages were changed. For example, an inappropriate use of the **again** key used to generate a message including the phrase "cannot clone." This was replaced with the more sensible "Sorry, again doesn't work here." The phrase "user error" was removed from all messages, as were references to implementation details such as "nodes" and "children."

The information provided by **help** was changed to be more specific in certain places where it was terser than usual. Changes were also made along the lines mentioned above and to reflect the changes in vocabulary items.

2.4.4 Keyboard Layout

The keyboard provided with the new version of the Nu terminal is well suited to Etude's needs in several respects. It has a large number of extra function keys, many of which are double the width of the standard typing keys, and it is programmable. This let us assign any functions that we desired to any key that we wanted, and to make changes in these assignments quickly. Since the keytops on the keyboard were engraved with the names of special functions appropriate to the original use of the keyboard (the keyboard is the same one used by the Artificial Intelligence Laboratory's Lisp Machine), stick-on labels with the name of the key were placed on the keytops. The keyboard that was used is diagrammed in Figure 2-3 on the next page.

Notice that verbs are generally on the left hand side of the keyboard, modifiers (including the numbers) are in the middle, and nouns are on the right. This follows two principles of keyboard design described in McCormick's book on human factors [66]: keys that have similar functions should be grouped together, and they should be

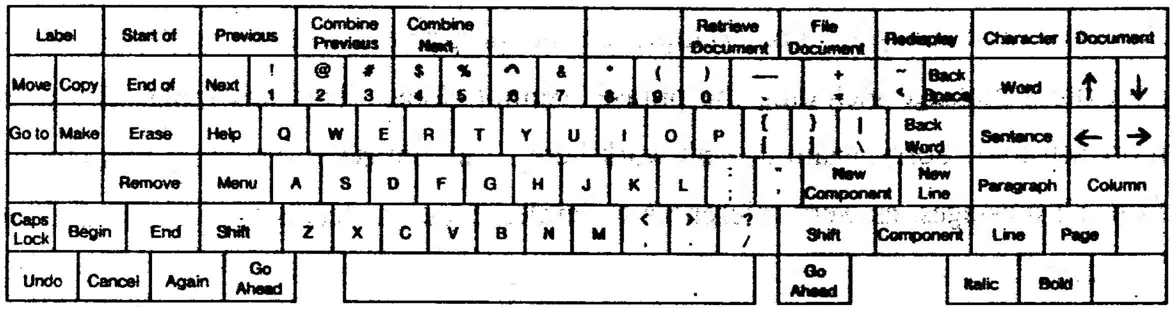


Figure 2-3: Etude Keyboard for Experimental Version

arranged to take advantage of the sequence in which they are used (pp. 291–92). In addition, "dangerous" keys (such as the I/O commands, which cannot be undone) are placed in out of the way positions. It is common to hit a key adjacent to a frequently used key, especially in the direction of overreaching [12]. The keyboard was laid out so that these errors would not cause terrible errors. In particular, the **go ahead** keys were placed under the shift keys. Striking the **go ahead** key instead of the shift key results in an unshifted character in the text, and the harmless message "Nothing to go ahead with." One would certainly not want a dangerous key in this position, nor a key like **help** which displays a lot of additional information on the screen.

Many computer systems that have large keyboards with numerous special keys suffer from having small, standard size special keys. The size of the keys limits the length of the function name that can be associated with each key to five (occasionally six) letters. Since many of the words from which these functions are derived are longer than six letters, the user is often confronted with a keyboard full of cryptic abbreviations. This almost completely undercuts the advantage gained by the special keys. An overriding concern in Etude's keyboard arrangement was that the full name of each function should be shown on the key. For example, **previous** is placed above **next** because **next** can fit on the small key on the second row, while **previous** needs the large key on the top row in order for the entire name to be displayed. The layout of verbs and nouns on either side of the keyboard was affected by the same criterion.

The decision to use labels on top of the already existing keys was necessitated by the circumstances. It would have been more desirable to have custom built keys with the function names already included. Some keyboards use so-called "reledgible keytops," which are plastic covers intended to fit over each key. A paper label can be stuck inside the cover before it is placed over the key. This provides a more durable container for a changeable label and would have been an improvement over the experimental arrangement; unfortunately, these keytops were available only for small, standard size keys and not for the larger ones. Color coding the keys—one color for the standard keyboard, another color for verbs, another for modifiers, and another for nouns—would also have been desirable [12].

There was no question of rearranging the basic keyboard. Any advantages of a redesigned keyboard would be completely outweighed by its unfamiliarity to the entire user population. The "inefficiency" of the current keyboard is also not very great. A greater source of increased typing efficiency is the elimination of keystrokes. For instance, automatic word-wrap eliminates the need to hit the **new line** key and takes very little retraining [55].

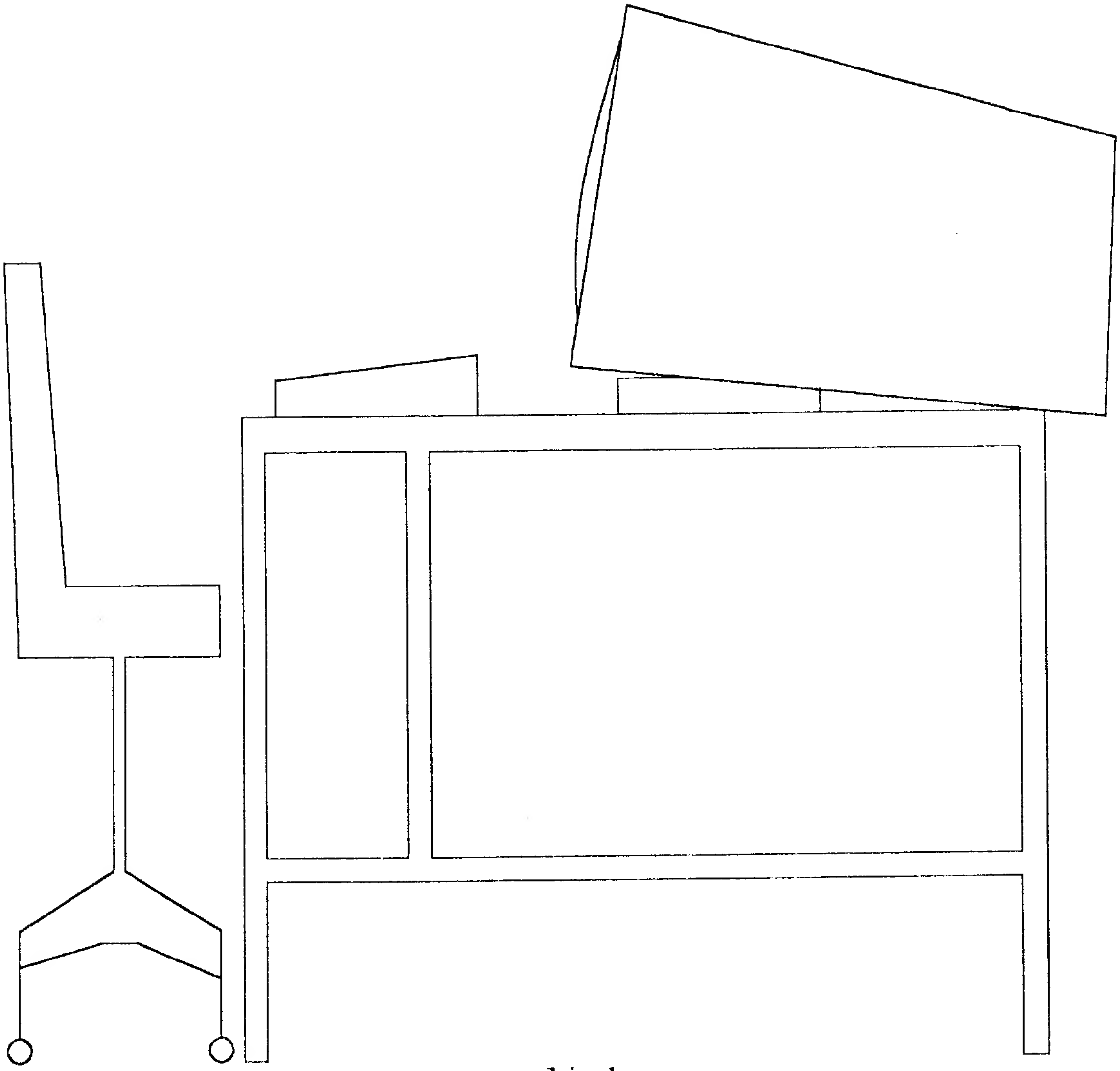
2.4.5 Workstation Design

Workstation design was the factor over which I had the least control, since I was constrained by the hardware associated with the Nu and the furniture available in the room where the workstation was located. Cakir, Hart and Stewart [12] provide a very full list of workstation design guidelines, a few of which will be discussed here.

From a human factors standpoint, the major problem with the Nu terminal is its display. The flicker on the screen is noticeable, though not as bad as on the display of the Xerox Alto computer. Even more annoying, the display will sometimes "jump" repeatedly, making it very hard to read the screen. Both of these problems contribute to eyestrain over an extended period of time.

Personal experience with the terminal alleviated one problem with the workstation setup. When the Nu is placed flat on the table, the top of the display is too low to be used at its 80 degree angle. Therefore, we stuck a Boston phone book underneath the front part of the Nu in order to prop it up, changing the angle and increasing the height of the top of the screen. This did not improve things enough, as my own neck pains showed. Therefore, I replaced the Boston phone book with a thicker MIT course catalog, and pushed it back farther to prop the terminal up higher. This proved to be satisfactory, with the angle of the screen reduced by 5 degrees. Figure 2-4 shows the final workstation setup.

Dimensions involving the desk top and keyboard are not ideal, but do not seem to have much of an effect in short term usage (the subjects usually spent no more than three hours in front of the terminal, including breaks). The thickness of the keyboard from the base to the home row of keys is 3 1/8 inches (80 mm), much greater than the "acceptable" 50 mm or the "preferred" 30 mm figures. At 27 1/2 inches (700 mm), the desk height is less than the recommended 720 to 750 mm, but the height of the keyboard above floor level (30 5/8 inches, 780 mm) is greater than this recommended interval.



— = 1 inch

(Width of chair not drawn to scale)

Figure 2-4: Workstation Setup

2.4.6 Display Layout

The display layout of the experimental version differs somewhat from that of the first version, as shown in Figures 2-5 and 2-6. The most noticeable change is the change in font; the new font is much smaller and more attractive than the old font. The smaller size enabled a full page of text to actually be displayed on the screen.

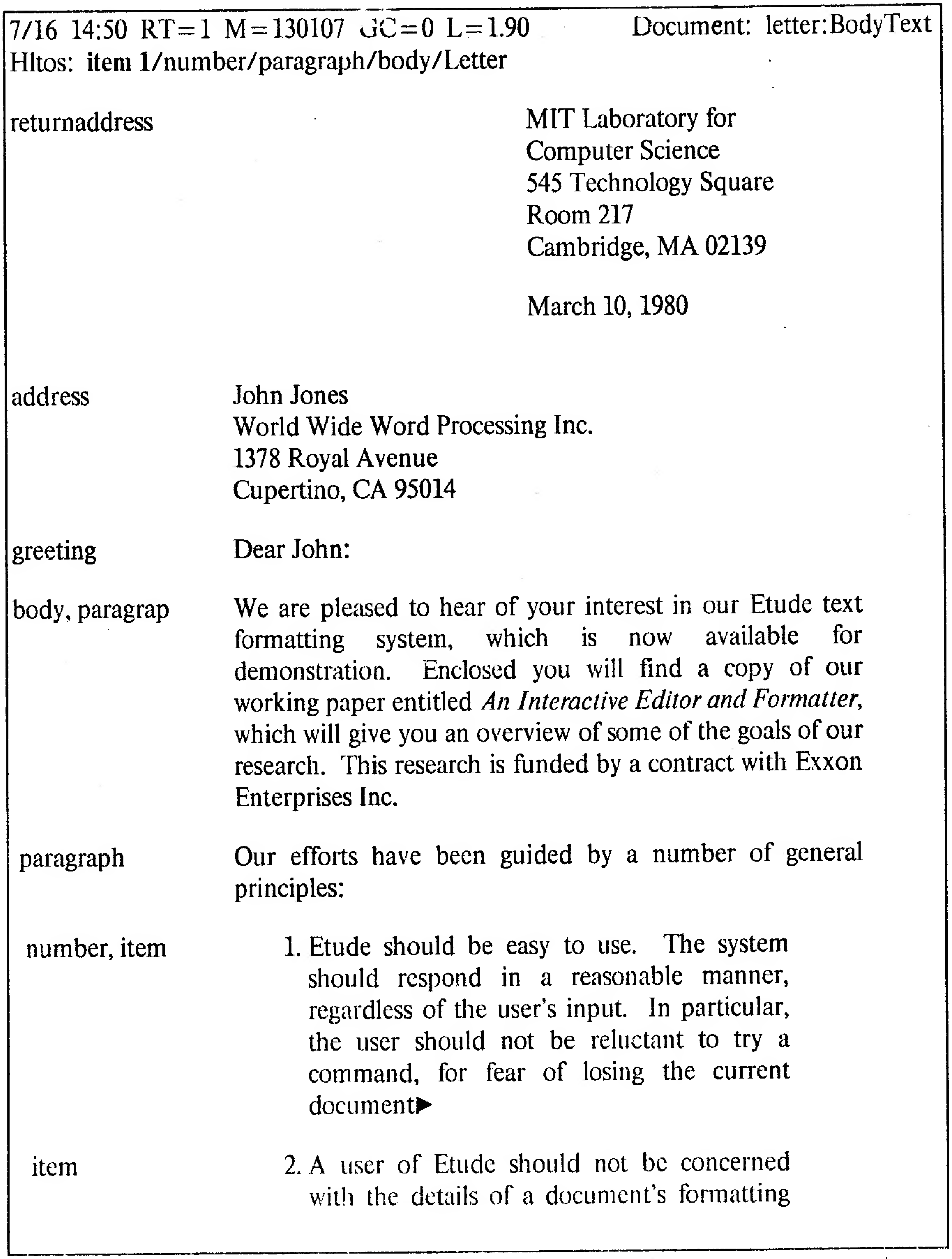
I also redesigned the interaction window. System information such as number of garbage collections and heap size (the figures indicated by GC and M) was eliminated, while the system load figure L was augmented with information for interpreting it. The "response time" indicated on the top of the screen could be "very good," "good'," "fair," "poor," or "very poor," depending on the value of L. The boundaries were determined in an ad hoc manner, based on experience with Etude. The time was changed from the 24 hour clock to a 12 hour clock using am and pm. In the right hand corner, the subdocument name was replaced with the name of the document.

On the second line, the ordering of components was reversed, so that the current component is now on the right hand side of the line. Changes to this line are reflected by the line getting longer or shorter, with the differences coming at the end of the line. This eliminated the need for highlighting the current component (represented by boldface in Figure 2-5).

2.4.7 Writing a Tutorial

The first draft of the Etude tutorial was written by Eric Munro, an undergraduate student who had experience in training people to use typesetting systems. I changed much of this draft in producing the final version, shown in full in Appendix A.

The basic goal of the tutorial was to provide a self-teaching facility, similar to that provided in the tutorial for the EMACS text editor [108]. The tutorial introduces various



7/16 14:50 RT=1 M=130107 GC=0 L=1.90 Document: letter:BodyText
Hltos: **item 1**/number/paragraph/body/Letter

returnaddress

MIT Laboratory for
Computer Science
545 Technology Square
Room 217
Cambridge, MA 02139

March 10, 1980

address

John Jones
World Wide Word Processing Inc.
1378 Royal Avenue
Cupertino, CA 95014

greeting

Dear John:

body, paragrap

We are pleased to hear of your interest in our Etude text formatting system, which is now available for demonstration. Enclosed you will find a copy of our working paper entitled *An Interactive Editor and Formatter*, which will give you an overview of some of the goals of our research. This research is funded by a contract with Exxon Enterprises Inc.

paragraph

Our efforts have been guided by a number of general principles:

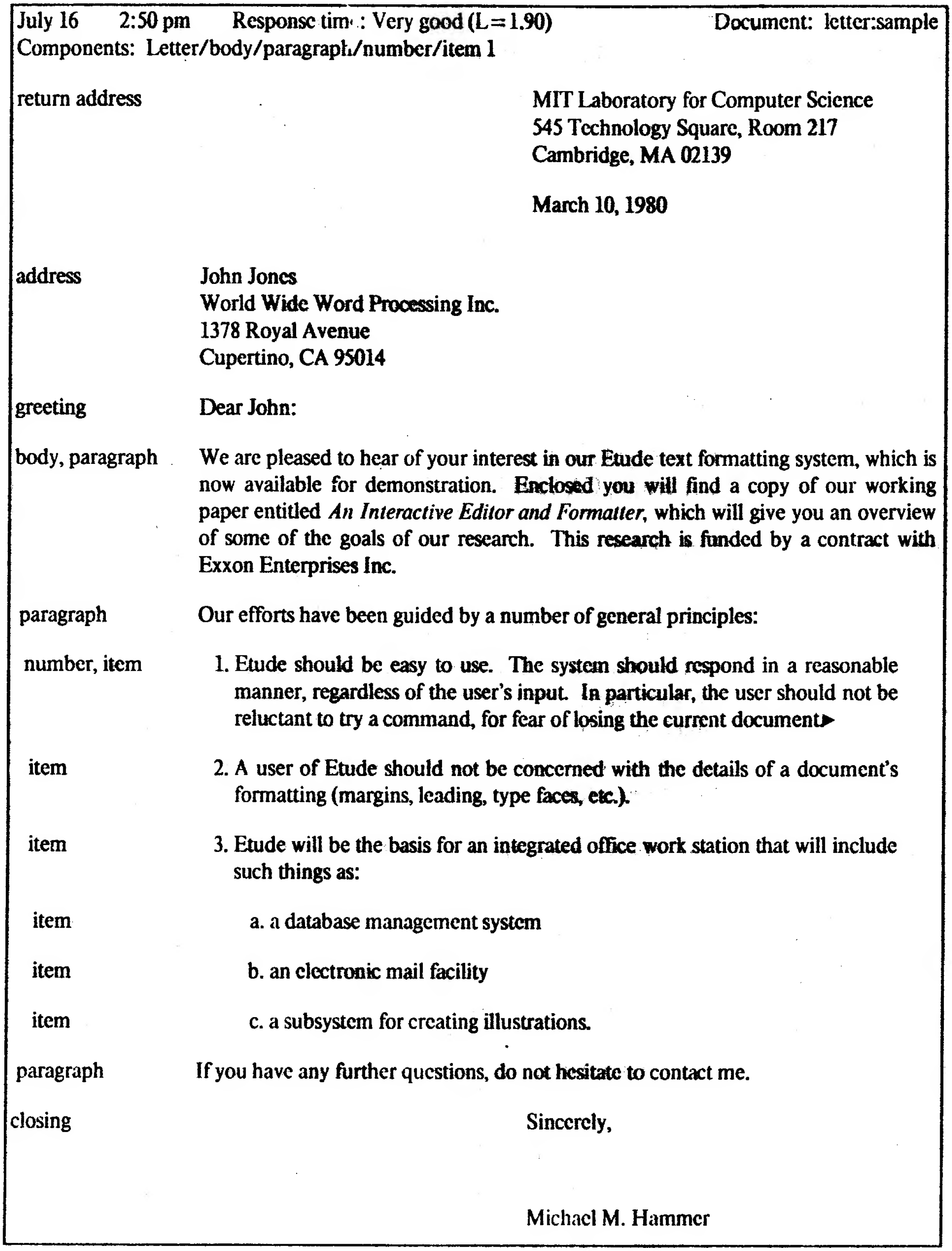
number, item

1. Etude should be easy to use. The system should respond in a reasonable manner, regardless of the user's input. In particular, the user should not be reluctant to try a command, for fear of losing the current document▶

item

2. A user of Etude should not be concerned with the details of a document's formatting

Figure 2-5: Etude Display Layout—First Version



July 16   2:50 pm   Response time: Very good (L=1.90)   Document: letter:sample
Components: Letter/body/paragraph/number/item 1

return address   MIT Laboratory for Computer Science
545 Technology Square, Room 217
Cambridge, MA 02139

March 10, 1980

address   John Jones
World Wide Word Processing Inc.
1378 Royal Avenue
Cupertino, CA 95014

greeting   Dear John:

body, paragraph   We are pleased to hear of your interest in our Etude text formatting system, which is now available for demonstration. Enclosed you will find a copy of our working paper entitled *An Interactive Editor and Formatter*, which will give you an overview of some of the goals of our research. This research is funded by a contract with Exxon Enterprises Inc.

paragraph   Our efforts have been guided by a number of general principles:

number, item   1. Etude should be easy to use. The system should respond in a reasonable manner, regardless of the user's input. In particular, the user should not be reluctant to try a command, for fear of losing the current document▸

item   2. A user of Etude should not be concerned with the details of a document's formatting (margins, leading, type faces, etc.).

item   3. Etude will be the basis for an integrated office work station that will include such things as:

item   a. a database management system

item   b. an electronic mail facility

item   c. a subsystem for creating illustrations.

paragraph   If you have any further questions, do not hesitate to contact me.

closing   Sincerely,

Michael M. Hammer

Figure 2-6: Etude Display Layout—Experimental Version

ideas and encourages the user to try things out as he goes along. It emphasizes at the beginning that the person is looking at a copy of the document, and that the original cannot be hurt by anything he might do in the tutorial.

While the tutorial is intended to be self-contained, it is not intended to be the sole source of information for a new user. Efforts to train people entirely with written material often encounter problems. For example, a user will often misunderstand one portion of the material. Without someone available to answer questions, this misunderstanding may sidetrack the user badly, considerably delaying the completion of the training period [57]. Having someone around to answer questions can also alleviate problems caused when the user has a slightly faulty model of the system [9].

How does one introduce a system such as Etude to computer-naive people? Several principles were followed in writing the tutorial:

- Emphasize the function of the machine. The user needs to understand what the machine does before understanding how it does it.

- Use analogies to familiar concepts where appropriate. If there are minor variations from the analogy, make sure that they are explained.

- Be very careful of the terminology that is used. Make sure that any new concepts, new words, or new meanings of words are explained to the user.

- Describe the system as it actually works, not as it should work.

These principles are illustrated in the following overview of the tutorial.

In the first chapter, we introduce the user to Etude, telling him that "Etude is a machine that lets you type up written material ... and see the material displayed in quality form." The user is told that Etude displays a *copy* of the document on the video screen, and that he can make changes easily. The function of printing a copy of the document is not emphasized, since the experimental version has no command for printing a document.

It is then emphasized that the user controls Etude, referring to Etude as "your slave." This emphasis has been recommended by Kennedy [53] and Shneiderman [98], and can help avert what Bott [9] calls the "commander-commandee" problem: when told about system "commands," naive users may think that the computer is giving them the commands, rather than the other way around. The tutorial concludes with an exhortation to try things out as they are introduced.

The second chapter introduces the user to the cursor and ways to move it. The arrow keys are mentioned, and the example of **go to next page** is motivated by the need to get to the next page of the tutorial, rather than being presented as "a command to use." This example introduces the user to the idea of English-like commands. This example is generalized in chapter 3, where the layout of the keyboard is described and the **erase** key is introduced. The user is also told about the response line and the use of **go ahead** and **cancel**.

Chapter 2 also addresses one of the experimental version's primary problems—its slow response time. It mentions that Etude tries to let the user know what to expect it terms of response time. The user is told that though he need not wait for Etude, he can stop and let Etude "catch up." Phrases like this and the previous "your slave" remark are intended to emphasize the person's control over Etude and to avoid feelings of being awed or intimidated by the machine.

Before any more commands are introduced, the user is told about the **help** and **undo** keys. Since Etude does not always clear all the help information from the screen, the **redisplay** key is also introduced—another example of dealing with the system as it is, not as it should be.

While **undo** is introduced as a way to correct mistakes, it is also presented as a way to experiment with the system. Throughout the tutorial, the user is asked to try a command and then **undo** it. This encourages experimentation and familiarity with the **undo** key, which should contribute to more natural usage.

At this point in the tutorial, we digress to the theory of Etude, introducing the user to the idea of components, using the example of a letter. This lets future examples include the names of components as well as items on special keys.

Chapter 6 tells the user how to insert text into the document. The term "text" is explained, and contrasted to the idea of a document's *structure.* Bott found when naive users were given the word "text" without any explanation, they thought it referred to the content of a textbook. They did not think of "text" as referring to the content of any document, as the term is typically used in computer systems. This illustrates a word whose meaning as a common word in computer terminology is different from its meaning as an infrequent word in regular English usage.

The user is told that he simply has to type text where he wants it, and that Etude will move the existing text over to make room for the new material. Again, it is mentioned that Etude may have trouble keeping up with the user; Etude is in fact very inadequate when it comes to keeping up with a typist.

The next chapter tells the user how to specify component names. The **menu** key is introduced here, instead of the earlier section on user aids, because this is the only practical case in which a menu is available in the experimental system. Though the **menu** key is *intended* to be universally available (as **help** already is), it cannot be introduced in that manner when in fact its use is quite limited.

Since many of Etude's commands involve regions, this is introduced before other editing commands are mentioned. The user can practice defining regions by using the **erase** command. After regions are introduced, the **move** and **copy** commands follow naturally in the next chapter.

Chapter 10 describes how to type in components by using the **begin** and **end** keys. The **new component** key is also described. Chapter 11 concludes the tutorial by showing how to use an "empty document" to create a new document. An empty document contains a basic editorial structure for the particular document type, but has no content. The user can then **go to** each component and type the appropriate text. This serves as a further memory aid for the new user. The **retrieve document** and **file document** commands are also introduced in the last chapter.

If time is available, an iterative process for improving the tutorial such as that recommended by Al-Awar, Chapanis and Ford [1] is certainly advisable. Due to time constraints, the experimental pre-tests provided the only opportunity to get user feedback before the experiments began. These pre-tests revealed only minor problems, primarily reflecting the need to include more examples earlier in the tutorial; more detailed results are given in section 4.5 starting on p. 85. The pre-tests also showed that the tutorial was a bit too long. The tutorial was shortened by removing the description of the **make** and **remove** commands.

One problem with the tutorial escaped notice during the pre-test, but showed up in the experiments. The end of the tutorial should have been rearranged to put less emphasis on using the **begin** and **end** keys for typing formatted text and more emphasis on using the empty document, since the latter is more frequently used by novice users. More extended pre-testing might have detected this problem.

Some features were ignored in the tutorial, such as the **combine previous, combine next, label**, and **anchor** commands. **Anchor** is not even included on the keyboard, but is only available through a special sequence of characters unknown to the naive user. The confusing and error-prone abbreviation scheme is also omitted, as is the idea of a search string object. The **component** key is not mentioned, and this is probably a mistake; using **next component** might be better than the **go to** <component name> scheme.

2.4.8 Reliability and Instrumentation

There are three distinct ways in which Etude could "break," each with its own set of consequences and recovery procedure:

- An error in the Etude program itself could cause the program to halt.

- The Nu terminal could malfunction.

- The mainframe computer could crash.

Since each of these problems happened with much more frequency than would be tolerable in a truly functional system, backup facilities were required to minimize the amount of work that would be lost. In the experiment, only four of the twenty-one subjects were able to use Etude without encountering at least one of these malfunctions.

A system log facility was added to Etude. This recorded each keystroke, and also recorded error messages that were given to the user. After 100 keystrokes, the log would be timestamped and the current document written out to disk. All but the two most recent versions of the current document were deleted from the disk. This appeared to be a reasonable tradeoff between complete safety and low cost. If the mainframe went down, Etude could be restarted after the computer came back up with only a small loss of work. If the Nu malfunctioned, the terminal would be reset through a multi-step procedure, after which Etude could continue without being restarted and without a loss of work.

Errors in the Etude program are the easiest to recover from. Before Etude halts due to an irrecoverable error, it saves the current document and closes the log file, inserting a note that Etude had broken. Thus, a malfunction due to a bug in Etude would result in no loss of work, though Etude would have to be restarted.

2.4.9 Remaining Problems

Since the experimental version of Etude was developed by making modifications to the user interface of the first version, some of the more deeply rooted problems with the first version could not be fixed. Time constraints also hampered efforts to make changes. Though the experimental version of Etude satisfies a large number of guidelines for user interface design, it is important to point out its failings as well. All of these problems are being worked on in the new version of Etude [49].

The major problem with the current version of Etude is that it is not able to keep up with a typist. Most people agree with Miller [70] that the response time to a typewriter keystroke should be almost instantaneous, not exceeding a tenth of a second. A fast typist might have to wait several seconds for Etude to display newly typed text on the screen; even slow typists have to wait in many cases.

Several problems contributed to the slow response time. A major factor was that instead of running on a stand-alone computer as originally intended, Etude was running on a mainframe computer connected to a bit map display terminal through a 9600 baud line. The mainframe was not connected directly to a terminal, but to a stand-alone computer (the Nu) whose UNIX operating system was running a virtual terminal interface program. All these connections slowed down the response time of the system, even when the mainframe was not heavily loaded. When the mainframe was heavily loaded, as on summer weekday mornings and afternoons, Etude was intolerably slow.

Other problems were due to poor design decisions. As the name suggests, Etude was a study in building an office tool; in this case, it was our first attempt at building such a tool. The prototype was not intended to be anything but a demonstration tool, so efficiency was not high on the list of design criteria when the system architecture was devised. The CLU compiler used for the first version of Etude also did not provide a great deal of help in producing optimized code. Without improvements that were made in the CLU compiler used in the experimental version and efficiency improvements that were made in the virtual terminal interface for the new 68000 based Nu, Etude would probably have been too slow to evaluate at all.

Though user aids are provided, their use should be more extended. **Undo** should be able to backtrack further than one operation. **Help** should be able to provide more detailed information to those users who need it, through a query-in-depth facility [30, 89]. The **menu** key should be useable at any time. These goals were all present in the original specifications, but were not implemented in the experimental version.

An experiment by Baker and Goldstein [2] indicates that only currently relevant items should be displayed in a menu. Etude follows this guideline in some areas but not in others. If a user types the beginning of a component name and then presses **menu**, only the items that start with what he has typed so far are included. However, if the user presses **menu** when using the **go to** command, the menu will include all of the possible components in the given document type. If the user chooses a component that does not exist in the current document, he will get an error message after the command has been completed. Etude should be aware of the types of components that are actually a part of the current document and display only those components in situations such as this.

As mentioned earlier, the current implementation of automatic completion is confusing and error-prone. A better scheme might be for Etude to automatically provide the completed name whenever it can, without waiting for the user to press **go ahead**. This relieves the user from the burden of remembering the correct abbreviation. The completed name could change as the user types in more characters, or the user could use the command line editing facilities to fix an incorrect completion.

One of the trickiest problems in building an interactive editor and formatter is the fact that one physical location on the screen may map onto many logical locations within the document. Put another way, one position in the outward appearance can correspond to several different positions in the editorial structure. A cursor positioned at the end of a list might be positioned inside or outside of the last item in the list. If the list also is at the end of a paragraph, the cursor might be inside or outside the list as well. Text typed at the current cursor position will be formatted differently in each case.

The experimental version of Etude attacked this problem by displaying the names of the components in the format window and by displaying the editorial structure at the current cursor position in the interaction window. Unfortunately, the latter information is often more useful than the former but is displayed in a remote corner of the screen, far from the user's normal focus of attention. A better solution would be to have the name of the current component highlighted in the format window, which is much closer to the user's focus of attention. It should be emphasized that this mapping problem is not unique to Etude but is faced by any interactive editor and formatter. JANUS [15], a system under development by an IBM research team, attacks this problem by using two displays. A conventional terminal is used to display the editorial structure, and a graphics terminal is used to display the outward appearance.

Chapter Three

Criteria for Ease of Use

Nearly every new computer system claims to be easy to use, and there is nothing new about the widespread nature of these claims [3]. The obvious question to ask when this claim is made is, "What do you mean by easy to use?" In other words, what ease of use criteria are being used?

The general criteria for ease of use that are used to evaluate Etude are closely related to ease of use criteria described in the literature. Each of these general criteria must be developed into specific criteria suitable for forming experimental hypotheses. This also requires that a subject population and a point of comparison be chosen. The development of these general and specific criteria is described in this chapter. The choices of data to be collected and tests to be used are deferred until the next chapter, since these choices interact with other details of the experimental design.

3.1 The General Criteria

When someone claims that a system is easy to use, several questions can be asked in order to qualify the claim:

1. Can the system be learned quickly?

2. Can it be used efficiently once you've learned it?

3. Does it make the user feel at ease?

4. Do people enjoy using the system?

5. What is the population of users for whom this system is easy to use?

6. When you say your system is easy to use, what are you comparing it to?

These questions summarize the concerns that have been dealt with most often in the literature on ease of use criteria. The first four questions deal with different areas of ease of use, while the latter two serve to further qualify all of these areas.

In this study, each of the first four questions has lead to a general criterion for ease of use. These were the criteria mentioned in the introduction:

1. Ease of learning,

2. Ease of use once learned,

3. The anxiety factor,

4. User attitudes.

The population being considered consists of secretarial workers who are computer-naive. The term computer-naive is used here to refer specifically to people who have not used a computer text processing system before. Comparisons are being made with the tool currently used by this population—the typewriter.

3.1.1 Etude and Its Users

The general criteria for ease of use were developed through consideration of the requirements of the people who will use advanced office systems such as Etude. These users may be either clerical or managerial workers, but in either case they will not necessarily have any experience with using computers.

If a system is not easy to learn, it will not be used. Management will be reluctant to invest a large amount of time in the training of clerical workers, especially with the rapid turnover in this field. Managers will invest even less time in any attempts to learn to use the system themselves.

While ease of learning is the first hurdle that must be cleared for an advanced office system to win acceptance, ease of use once learned is at least as important. If a system is cumbersome to use it will either be circumvented or it will be used in its own inefficient way. Neither of these outcomes is desirable.

User satisfaction with the system is as important a goal as user performance [64]. In addition to the previously mentioned anxiety factor, user attitudes towards the system provide a straightforward indication of user satisfaction.

Because ease of use is multi-dimensional, a system may satisfy some of these criteria without satisfying others. Several authors have recognized this problem, including Miller [71] and Gebhardt and Stellmacher [32]. The latter considered the tradeoffs between various design criteria in detail, and concluded that the tradeoff between simplicity for the casual user and flexibility for the experienced user is especially difficult to resolve. Certainly there are several systems that are either easy to learn or easy to use, but a successful office system must meet both of these criteria and satisfy its users in the process.

3.1.2 Choosing a Subject Population

Most ease of learning experiments have used computer-naive users for several reasons. Computer-naive users should be the most difficult population to teach, because they have to be introduced to the idea of using a computer-based tool as well as to the tool itself. This population poses a more stringent test for ease of learning than would a more computer-experienced population.

In addition, the use of people without prior computer experience avoids the problem of transfer effects; that is, the transfer of knowledge a subject has about one system over to another similar system. The transfer can be beneficial where parts of the systems are identical, but harmful where they differ, especially when the differences are small or subtle. In either case, the transfer adds another source of variance to the experiment which could obscure the effect that is being measured.

What constitutes a computer-naive worker? Answering this question leads to the conclusion that this is a transitional period for experimentation with computer text processing systems. In metropolitan areas such as Boston, it is hard to find people who have never used a computer. Besides the issue of "hidden" computers in automobiles, appliances, and toys, most of the major banks have 24 hour automated tellers complete with video screens. Many workers have also been exposed to simple data entry devices.

At the moment, there is still a substantial number of secretarial workers who have never used a word processor, but the proportion of workers who fall in this category will be decreasing. These next few years, then, might be the last chance that experimenters will have to easily find subjects who are experienced in secretarial work but who are computer-naive to the extent of not having used a word processing system.

The decision to use computer-naive subjects rather than computer-experienced subjects was made primarily for the reason of constructing a more stringent measure of ease of use. The fact that this type of subject is becoming increasingly rare was a secondary consideration.

Avoiding transfer effects makes life easier for the experimenter. If only one level of computer experience is included in the subject population, using computer-naive subjects leads to more easily generalized results. The problem of transfer effects can be controlled, however [56, 57]. There will probably be a shift towards using computer-experienced people as subjects as they become more representative of the general user population. To increase generality at the cost of complexity, varying levels of experience can be included in an experimental study.

3.1.3 Choosing a Point of Comparison

The decision to use a typewriter rather than another word processor as a point of comparison may seem questionable, but there are several reasons behind this choice.

1. Using a typewriter is likely to give us the most stringent test we could want in terms of the anxiety factor, since the subjects have been using a typewriter for most or all of their working lives.

2. Using a typewriter avoids complexity in the experiment that would arise from the need to teach two different systems.

3. For simple tasks such as typing letters, a typewriter is probably still the most efficient tool for the task.

Once Etude has been implemented in a stand-alone environment with an appropriately rapid response time, it would definitely be worthwhile to compare Etude with other text processing systems. This is discussed further in Chapter 6.

3.2 Ease of Learning

A straightforward way to measure ease of learning is to measure the length of time it takes people to learn how to use the system. Two possible choices for a metric are:

- Measures of central tendency, such as the mean or median time required for subjects to learn the system.

- The proportion of subjects who can learn to use the subject in a given amount of time.

Since these metrics are quite similar, the choice between them is usually determined by details of experimental design. The first choice happens to be the same as one of Miller's ease of use criteria [71].

A major question when using the above criteria is the meaning of the term "learn to use the system." There are various levels of proficiency that could be measured in this way:

1. The time required to become acquainted with a system so that basic tasks can be performed, though not necessarily with great speed.

2. The time required to learn a system well enough to be proficient at basic tasks.

3. The time required to be proficient at basic tasks and capable of performing advanced tasks.

4. The time required to be proficient at basic and advanced tasks.

This leads us to the problem of defining terms such as "basic tasks," "advanced tasks," and "proficient." The notion of "capable" seems fairly clear, simply indicating the user's ability to get a task done.

The goal in this study was to measure the amount of time it would take people to learn enough of Etude so that they could carry out some useful work. The simplest task that is useful, familiar, and involves most of the basic concepts of Etude is the task of typing and correcting a letter. Letters are the most familiar type of document that exploit the use of formatting knowledge associated with editorial structure. A return address has a certain left hand margin associated with it, along with space to be left above and below it. The margins and space requirements are different for other components, such as paragraphs.

An alternative measurement of ease of learning would be to measure learning rates, as was done in Roberts' core learning experiments [90]. Roberts created a basic training method which could be used with various text editors, and included several quizzes with the training material in order to measure the number of tasks learned per unit time. While learning rates are a less attractive measure than total training time for the purposes of this study, Roberts' method did allow for comparison of total training time with other editors. Therefore, serious consideration was given to using her training method. Several factors were considered:

1. While Roberts' methodology allows various editors to be compared, it is not powerful enough to detect any but the crudest differences between editors. In Roberts' thesis, the only distinction that could be made between learning rates was that TECO, a notoriously difficult to learn text editor, was indeed harder to learn than the other three editors in the study. While it is encouraging to see experimental verification of commonly held beliefs, it is doubtful that Roberts' methodology is capable of making the finer distinctions among editors of more contemporary origin.

2. Roberts' teaching method was intended for use with text *editors*. While it can be extended to apply to interactive editors and formatters, it is doubtful that the teaching method would be as effective as one especially designed for such a system.

3. In a related problem, Roberts' quizzes emphasize the editing task. They gives little attention to the typing task, much less the task of typing a formatted document. The capabilities measured by these quizzes thus do not match the capabilities which are considered basic to the use of Etude.

4. While interspersing quizzes with training material is necessary for measuring learning rate, it probably increases the total training time beyond the minimum that would be necessary.

Considering these factors, I concluded that the likelihood that Roberts' method would introduce errors into the estimation of Etude's total training time was far greater than the likelihood that her method would result in a significant comparison. Thus, Roberts' training methodology was not adopted.

3.3 Ease of Use Once Learned

There are two major methods used to measure ease of use. One method is to measure the speed with which a user can use a system. Measurements may be in the form of average time to complete a task or the percentage of a task completed within a given amount of time. Again, the choice between these two similar criteria usually depends on details of the experimental design. Roberts' study with expert users used the former choice; Ledgard et al.'s experiment [56] used the latter.

The other method is to measure the amount of errors produced by users; the fewer the number of errors, the easier the system is to use. Ledgard et al. used two different measurements of errors:

- The percentage of erroneous commands used.
- Editing efficiency, measured by subtracting the number of commands that resulted in a degradation of the text from the number of commands that resulted in an improvement of the text, and dividing the result by the total number of commands issued.

Roberts used a measurement of the percentage of time an expert user spent in making and correcting errors.

This study does not use a measurement of errors, but relies strictly on measurements of speed of use. This decision is based on the very nature of the user's interaction with Etude. Users are encouraged to experiment with operations, since the **undo** key allows them to reverse the effects of the operation if the results are not wanted. We believe that this freedom to experiment is a major advantage in Etude's design. If we then proceed to measure errors or operations that degrade text, we would be penalizing Etude for encouraging experimentation. Without the use of videotape, it would be very hard to differentiate between intentional experimentation and actual mistakes.

The quantitative measurement, then, is the amount of time it takes a user to create and make corrections to a letter after he has gone through the training period. This is a measurement of ease of use for novices—that is, users who have completed the first and most rudimentary part of the learning process. A better measurement of ease of use would involve more skilled users. Actual system users do not remain novices for very long.

Another disadvantage with this measurement is that ease of use does not become a major advantage for Etude until the documents are longer than simple one page letters. Nevertheless, it is not unreasonable to expect that even the newest users of a computer text processing system should be able to edit a letter on the computer faster than they can retype it using a typewriter, though typing in a letter may be no faster. If this is not the case, then Etude is not easy for novices to use.

The question of using novices exclusively will be discussed further in the chapter on experimental design.

3.4 The Anxiety Factor

As mentioned previously, a major goal in Etude's design was to reduce the anxiety factor often associated with using computer systems. Although much has been written about feelings of frustration, anxiety, and pressure while using computers, very few efforts have been made to measure anxiety associated with computer usage in a quantitative way.

One reason for the paucity of work in this area may simply be the lack of knowledge about an easy to use instrument that is expressly intended to measure a person's anxiety at a particular time, such as during the performance of an experimental task. This instrument is the State-Trait Anxiety Inventory (STAI), developed by Spielberger, Gorsuch, and Lushene [105].

The STAI contains two questionnaires. One measures *state anxiety*, which is the anxiety that is present in a particular situation. It is a transitory emotional state which varies in intensity over time. The other questionnaire measures *trait anxiety*, or a person's anxiety-proneness. The criterion that we have called the anxiety factor can be refined to the particular criterion of state anxiety. Readable descriptions of state-trait anxiety theory and the development of the STAI can be found in the psychological literature [11, 58, 104].

The STAI questionnaire for state anxiety is made up of twenty items, including ones like "I am tense," "I feel calm," and "I feel nervous." The subject marks one of four possibilities for each scale: "not at all," "somewhat," "moderately so," or "very much so." Each scale is scored from 1 to 4. For half of the items (such as "I am tense"), "very much so" receives a 4; for the other half (such as "I feel calm"), "not at all" receives a 4. The scores for each scale are added up to form the total score.

Most studies involving the STAI use it to measure anxiety in situations where anxiety is an independent variable; that is, when anxiety is perceived as effecting some other criterion. In this study, the STAI is being used to measure anxiety where it is a dependent variable; we want to determine if a subject's state anxiety changes when using Etude. A few studies have been made where the STAI is used to measure the effect of a computer system on anxiety.

Most of the work with the STAI and computer systems has been in the field of computer aided instruction, where the connection between anxiety and learning is often of interest. The relationship is complicated, but may be simplified by stating that subjects with low state anxiety will perform better in learning experiments dealing with difficult tasks than subjects with high state anxiety, but that the results are reversed when the experiment involves easy tasks [106]. One study gave the more direct result that providing feedback in a computer aided instruction system led to reduced state anxiety as measured by the STAI [41].

Walther investigated the effects of interface flexibility, terminal type, and subject experience on various performance factors related to the ease of use of a simple text editor [115]. One of these factors was state anxiety, as measured by the STAI. The results were not conclusive, due in part to the many variables involved in the study.

3.5 User Attitudes

A few studies have dealt with the question of user attitudes by sending questionnaires to users of a particular system, aimed at finding out what users liked and disliked about the system. While this is certainly a useful technique, the data that is collected is usually limited in the power of statistical tests that can be used with it, since in many cases the data is either dichotomous (yes/no, like/dislike) or ordinal, where data can be ranked in categories, but the differences between categories are not necessarily equal.

Statistics such as means and standard deviations that are used in many types of hypothesis testing require that data be available on an interval scale, where the differences between units are equal. One of the advantages of the STAI is that it measures data on an interval scale. Most of the techniques for measuring attitudes on an interval scale requiring a great deal of effort in questionnaire construction to ensure that the intervals are indeed equal.

There is a method for constructing questionnaires that retains the property of equal intervals but enables the experimenter to construct a questionnaire very quickly. The Semantic Differential (SD) has been used quite extensively over the past twenty years, and a great amount of literature exists on the theory behind it and methodological considerations involved in using it [103]. *The Measurement of Meaning* by Osgood, Suci, and Tannenbaum [80] is the basic book on the SD. Heise has written two important papers on methodological issues [45, 46].

3.5.1 Using a Semantic Differential

An SD is made up of a series of scales. Three sample scales are shown in Figure 3-1.

	extremely	quite	slightly	neutral	slightly	quite	extremely	
large	○	○	○	○	○	○	○	small
fast	○	○	○	○	○	○	○	slow
good	○	○	○	○	○	○	○	bad

Figure 3-1: Three Sample SD Scales

Each scale is anchored by a pair of bipolar adjectives [20] such as "large-small," "fast-slow," and "good-bad." The subject is then instructed to rate a particular word or concept on each scale. Scales are usually divided into seven steps, each of which is qualified by an adverb. A subject told to rank the concept "dinosaur" on the scales in Figure 3-1 might indicate that he considered a dinosaur to be extremely large, quite slow, and neutral with reference to being good or bad.

Attitudes measured by an SD generally fall into three categories: *evaluation*, measured by scales such as "good-bad," *potency*, measured by scales such as "large-small," and *activity*, measured by scales such as "fast-slow." Many studies have been done to derive scales that measure one particular category across a large number of concepts. Usually, an SD is made up of equal or near-equal numbers of scales that measure each of the these three categories. Each scale is scored on a scale of -3 to 3 (or 0 to 7). The scores from the individual scales are then averaged over each category for a final measurement containing three scores. In comparing attitudes towards different concepts, only the evaluative category can really considered to indicate "better" or "worse" attitudes; the other two categories indicate “different” attitudes of a particular nature.

Adjective pairs interact with the concept being measured. Mitsos [72] and others have shown that if a particular adjective pair is not perceived as being relevant to the concept being measured, the measuring capability of the scale is reduced. Some words may also have different meanings when applied to particular concepts. This may lead to a situation where a scale which usually measures one attitude category (such as potency) turns out to measure another (such as evaluation) for this particular concept. If two different concepts are being compared, this interaction can have particularly bad results. Thus, each study requires the construction of its own SD to ensure that such problems are avoided. Fortunately, this process is not difficult.

Lucas’ study of patients’ attitudes towards medical interviews conducted by a computer [61] provides a very helpful example of the use of an SD in measuring attitudes towards computer systems. He contrasts the process of building a more traditional attitude scale with the process of constructing an SD. There are a few more studies that have successfully used an SD to measure attitudes towards a particular computer-based system [28, 107, 115]. A modified version of Lucas’ strategy was used to construct the SD used in this study.

3.5.2 Construction of a Semantic Differential

The construction of an SD involves the selection of scales that will measure each of the three primary attitude categories towards the concepts involved in the study. In this case, the concepts are “Etude” and “typewriter.” The scales should be relevant to the given concepts and should not interact with any of the concepts being measured.

Following Lucas, I decided to use four scales for each category in the final SD. Scales for each category were collected from a number of sources. From this collection, eleven scales for evaluation and nine each for potency and activity were selected for a preliminary questionnaire, which asked the respondents to rank the scales according to their relevance to the given concepts. The four scales that were judged to be most relevant in each category were chosen for the final SD.

Scales were collected from the following sources:

- Tables 3 and 4 from Jackobovits [50]

- Tables 3 and 5 from *The Measurement of Meaning* [80]

- Studies 1 and 3 in Table 1 from Osgood [79]

- Table 2 from Lucas [61]

- Table 2 from DiVesta [21]

- Appendix A from Walther [115]

- Factors 1, 2, and 5 in Table 4 from Spiliotopoulos and Shackel [107]

After the scales were collected, the following algorithm was used to choose scales for the preliminary questionnaire:

1. Use all scales mentioned in either Table 3 or 4 from Jackobovits.

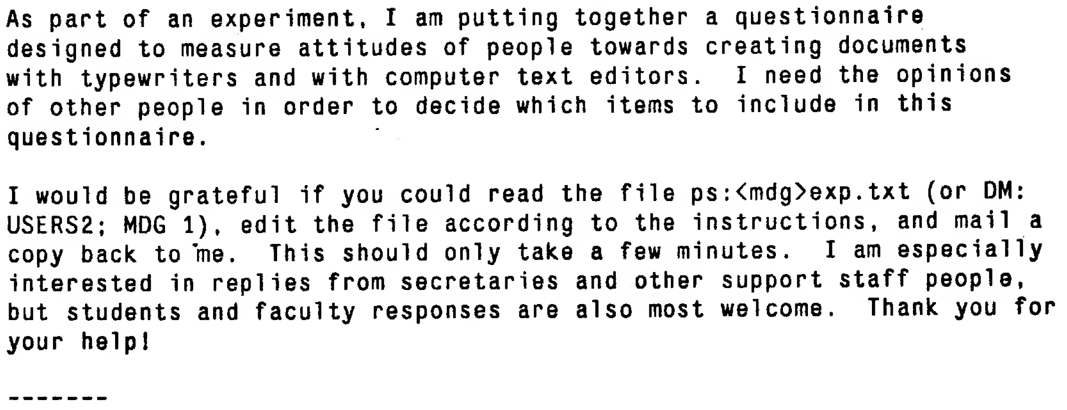
2. Use all scales mentioned in both Tables 3 and 5 from *The Measurement of Meaning.*

3. Use all scales from either *The Measurement of Meaning* or Osgood that were also ranked from 1 to 4 in Table 2 from Lucas.

4. As a special case, one scale used in DiVesta, Spiliotopoulos and Shackel, and Walther was chosen.

Figure 3-2 contains a message sent to the bulletin boards of the computer systems at the MIT Laboratory for Computer Science and Artificial Intelligence Laboratory. Figure

3-3 (two pages long) shows the preliminary questionnaire, which was contained in the files mentioned in Figure 3-2.



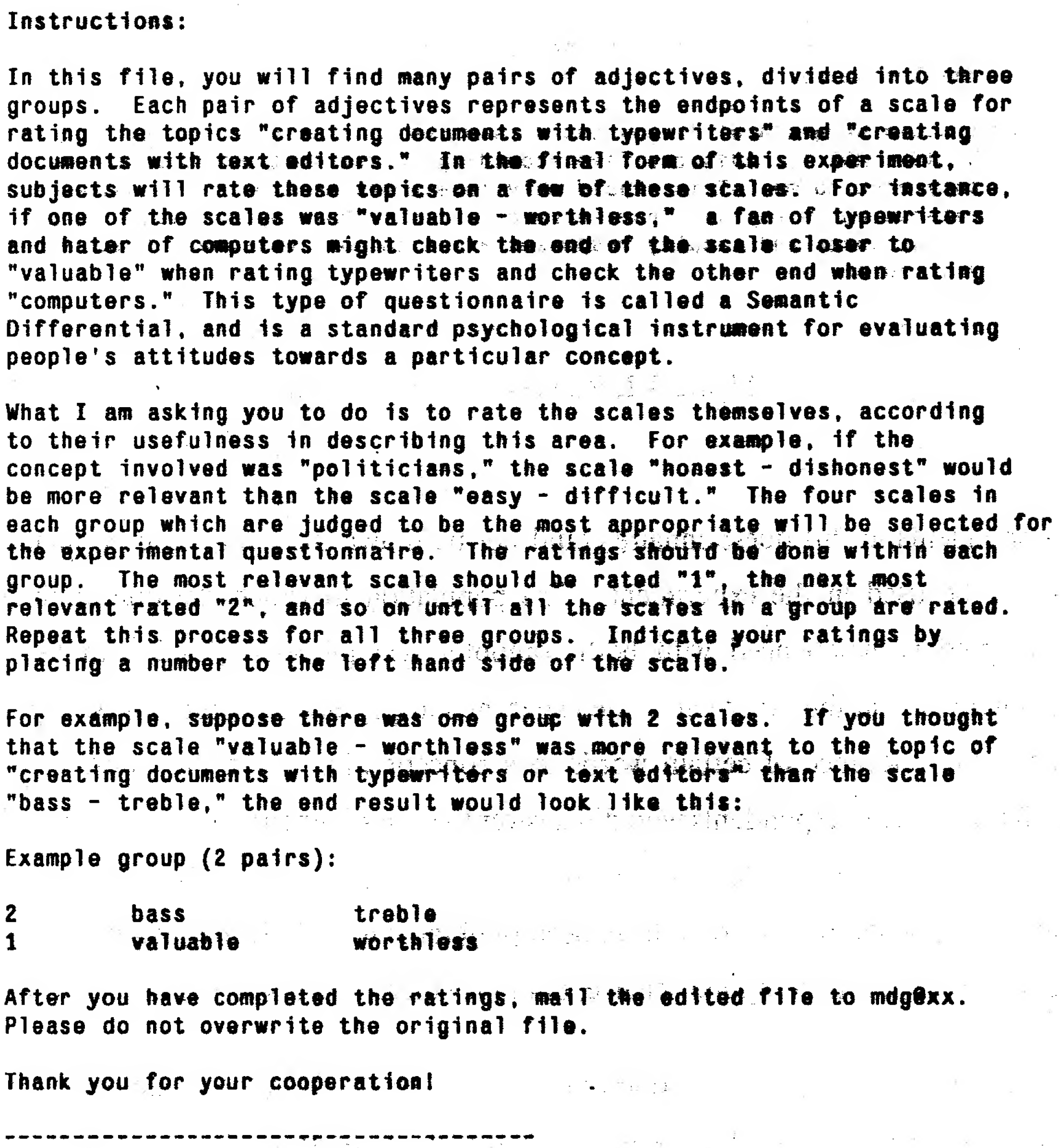
As part of an experiment, I am putting together a questionnaire designed to measure attitudes of people towards creating documents with typewriters and with computer text editors. I need the opinions of other people in order to decide which items to include in this questionnaire.

I would be grateful if you could read the file ps:<mdg>exp.txt (or DM: USERS2; MDG 1), edit the file according to the instructions, and mail a copy back to me. This should only take a few minutes. I am especially interested in replies from secretaries and other support staff people, but students and faculty responses are also most welcome. Thank you for your help!

Figure 3-2: Request to "Help a Student"

Respondents to this questionnaire could not be drawn from the same population that would be used in the experiment. The population to which the questionnaire was addressed is experienced with both computer text processing and with the use of typewriters. On the other hand, the experimental population could not be expected to judge the relevance of certain words to a concept with which they were completely unfamiliar. The respondents who did answer the questionnaire included undergraduate and graduate students, faculty, and support staff, representing a broad sample of the intended population. It would have been preferable to include the judgments of people less associated with computer science, but time constraints made this impractical.

Nineteen people returned this questionnaire; the results from the preliminary questionnaire are given in Table 3-1. Scales marked with an asterisk were selected to make up the SD. Two of the respondents did not give complete answers to the potency and activity scales, so those scales include the summed ranks of only 17 respondents. Kendall's coefficient of concordance *W*, a measure of the agreement among rankings such as these [52], was higher than the matching values in Lucas' study. The coefficients



Instructions:

In this file, you will find many pairs of adjectives, divided into three groups. Each pair of adjectives represents the endpoints of a scale for rating the topics "creating documents with typewriters" and "creating documents with text editors." In the final form of this experiment, subjects will rate these topics on a few of these scales. For instance, if one of the scales was "valuable - worthless," a fan of typewriters and hater of computers might check the end of the scale closer to "valuable" when rating typewriters and check the other end when rating "computers." This type of questionnaire is called a Semantic Differential, and is a standard psychological instrument for evaluating people's attitudes towards a particular concept.

What I am asking you to do is to rate the scales themselves, according to their usefulness in describing this area. For example, if the concept involved was "politicians," the scale "honest - dishonest" would be more relevant than the scale "easy - difficult." The four scales in each group which are judged to be the most appropriate will be selected for the experimental questionnaire. The ratings should be done within each group. The most relevant scale should be rated "1", the next most relevant rated "2", and so on until all the scales in a group are rated. Repeat this process for all three groups. Indicate your ratings by placing a number to the left hand side of the scale.

For example, suppose there was one group with 2 scales. If you thought that the scale "valuable - worthless" was more relevant to the topic of "creating documents with typewriters or text editors" than the scale "bass - treble," the end result would look like this:

Example group (2 pairs):

2	bass	treble
1	valuable	worthless

After you have completed the ratings, mail the edited file to mdg@xx. Please do not overwrite the original file.

Thank you for your cooperation!

Figure 3-3: Preliminary SD Questionnaire

Group 1 (11 scales):

beautiful	ugly
friendly	unfriendly
good	bad
happy	sad
heavenly	hellish
helpful	unhelpful
kind	cruel
mild	harsh
nice	awful
pleasant	unpleasant
sweet	sour

Group 2 (9 scales):

big	little
deep	shallow
hard	soft
heavy	light
high	low
large	small
long	short
powerful	powerless
strong	weak

Group 3 (9 scales):

active	passive
alive	dead
burning	freezing
fast	slow
hot	cold
known	unknown
noisy	quiet
sharp	dull
young	old

Figure 3-3: (Continued)

were: Evaluation = 0.67, $p < 0.001$; Potency = 0.53, $p < 0.001$; Activity = 0.75, $p < 0.001$.

Heise [46] recommends that the scales in an SD be mixed at random from all the different dimensions. Also, half of the scales should be reversed so that the "positive" end of the scale is not always on the same side of the page. These precautions discourage the formation of certain response sets which reduce the accuracy of

Factor	Scales	Summed Ranks	Rank Order
Evaluation	Beautiful — Ugly	139	8
	Friendly — Unfriendly	56	3*
	Good — Bad	95	5
	Happy — Sad	183	10
	Heavenly — Hellish	125	6
	Helpful — Unhelpful	45	1*
	Kind — Cruel	126	7
	Mild — Harsh	146	9
	Nice — Awful	88	4*
	Pleasant — Unpleasant	53	2*
	Sweet — Sour	198	11
Potency	Big — Little	97	5
	Deep — Shallow	110	8
	Hard — Soft	103	6
	Heavy — Light	106	7
	High — Low	127	9
	Large — Small	81	4*
	Long — Short	76	3*
	Powerful — Powerless	21	1*
	Strong — Weak	44	2*
Activity	Active — Passive	46	2*
	Alive — Dead	80	5
	Burning — Freezing	148	9
	Fast — Slow	30	1*
	Hot — Cold	129	8
	Known — Unknown	52	3*
	Noisy — Quiet	64	4*
	Sharp — Dull	108	6
	Young — Old	108	6

n = 19 for Evaluation; *n* = 17 for Potency and Activity

Table 3-1: Results of Preliminary SD Questionnaire

questionnaire measurements in general [122].

Following these guidelines, the scales selected through the preliminary questionnaire were randomly ordered. The scales to be reversed were randomly selected. The final questionnaire, complete with instructions for use with Etude, is shown in Figure 3-4. The instructions were identical for the typewriter version except for the substitution of the word "typewriter" for "Etude."

Questionnaire

The purpose of this questionnaire is to find out how you feel about using Etude. To do this, I'd like you to rate the word "Etude" in terms of several descriptive scales. For each scale, blacken in the appropriate circle to indicate how you feel about Etude. Here is a sample scale:

	extremely	quite	slightly	neutral	slightly	quite	extremely	
good	○	○	○	○	○	○	○	bad

If you thought that Etude was very good, you would blacken a circle on the side of the scale closest to the word "good." If you thought it was very bad, you would blacken a circle on the other end of the scale. Otherwise, you would blacken a circle towards the middle of the scale. Please be sure to make a rating on each scale.

Do not worry or puzzle over individual items. It is your first impressions, the immediate "feelings" about each scale, that I want. On the other hand, please do not be careless, because I want your true impressions.

	extremely	quite	slightly	neutral	slightly	quite	extremely	
noisy	○	○	○	○	○	○	○	quiet
helpful	○	○	○	○	○	○	○	unhelpful
large	○	○	○	○	○	○	○	small
awful	○	○	○	○	○	○	○	nice
friendly	○	○	○	○	○	○	○	unfriendly
known	○	○	○	○	○	○	○	unknown
slow	○	○	○	○	○	○	○	fast
powerless	○	○	○	○	○	○	○	powerful
active	○	○	○	○	○	○	○	passive
weak	○	○	○	○	○	○	○	strong
unpleasant	○	○	○	○	○	○	○	pleasant
short	○	○	○	○	○	○	○	long

Figure 3-4: Final SD Questionnaire

Chapter Four

Experimental Design

In the previous chapter, we presented four specific criteria with which to measure the ease of use of the Etude text processing system:

1. Training time required for users to learn how to create and edit letters.
2. Time required for novice users to create and edit a letter.
3. User's state anxiety as measured by the State-Trait Anxiety Inventory.
4. User's attitudes, especially evaluative attitudes, towards the system as measured by a Semantic Differential.

The users mentioned in these criteria are computer-naive secretaries.

This chapter describes the design of the experiment. Details of the experimental tasks, the experimental protocol, and the data analysis plan are presented. Changes made as a result of pre-testing are also discussed.

4.1 The Experimental Tasks

A major portion of experimental design involves determining what data is to be collected, and ensuring that the measurements that are used are representative of the specific evaluation criteria. In the case of the latter two criteria of anxiety and attitudes, the choice of data was derived directly from the criteria. Anxiety was measured by administering Form X-1 (State Anxiety) of the STAI to the subject after he has used a particular device. The SD was administered afterwards, using the Evaluation score to measure attitudes.

For the first two criteria, the refinement process was more complicated. In order to measure training time, typing time, and editing time, experimental tasks had to be developed on which the subjects could be timed.

4.1.1 Training Time

In the previous chapter, I alluded to some of the problems in determining criteria based on training time. Since the primary method of training in Etude is through the use of an on-line tutorial, the training time should include the length of time it takes for the subject to complete a tutorial. The tutorial should be limited to the basic skills needed to create and edit a letter.

How does one ensure that a subject knows how to create and edit a letter after finishing a tutorial? The obvious way is to have the subject create and edit a letter after finishing the tutorial; this serves as a test of the subject's knowledge. After this decision has been made, a number of detailed questions arise:.

- Should the time that it takes the subject to complete the test be included in the measurement of training time?

- How should the testing tasks of creating and editing a letter be constructed?

- When is the subject judged to have completed the test successfully?

- What amount of assistance should the experimenter give the subject during the training session and the test?

The Etude tutorial is in the form of a report; although letters are mentioned in the tutorial, the subject never does any manipulation with the components of the letter in a tutorial. For this reason, it was very doubtful that a subject would really know how to create and edit a letter after finishing the tutorial; as with all other aspects of learning Etude, practice would be required. Thus, the tasks of creating and then editing a letter were included as part of the training time. This is in agreement with the procedures followed in Roberts' study, where she noted that much learning takes place during these types of tests.

The task of creating and editing a letter was presented in a straightforward manner. The subject was presented with a letter to type. After the letter had been typed, it was proofread by the experimenter. Any mistakes noted in typing the letter were fixed, so that the letter is in fact "letter-perfect." After the letter was proofread and corrected, the subject was given a marked up version of the same letter, with corrections indicated by standard proofreading marks. The subject then made these changes to the version of the letter that was just typed. Again, the edited version of the letter was proofread by the experimenter to assure correctness. The total training time was measured from the time when the subject started to read the tutorial to the time that the last correction was made to produce a perfect copy of the revised letter.

There are two types of errors that the subject can make when working with Etude: errors in specifying the content of a document and errors in specifying the editorial structure of a document. The former are simply typing errors, all of which had to be corrected. The treatment of mistakes in editorial structure was more complex. The most important point in the training session was to ensure that the subject realizes his mistake so that he will not repeat it. Some of the mistakes that subjects can make in specifying the editorial structure cannot be undone by the methods taught to subjects in the tutorial; in this case, the experimenter undid the mistakes where appropriate. The tasks were judged to be completed if the outward appearance is readable; it did not have to be perfect, as was the case in the content. This question assumes more importance in the test for speed of use and will be discussed further in the next section.

This also brings up the question of the amount of assistance that the experimenter can give to the subject during the training session. Basically, the experimenter could answer any question posed by the subject in the training session. The subject was told that while the tutorial is largely self-explanatory, he should feel free to ask the experimenter any questions that he may have. During the training session, the experimenter was seated at a table in another part of the room, and was usually reading or writing during the session. This closely resembles an actual learning situation for many systems; new users can question more knowledgeable people, but since this might involve interrupting someone, the questions aren't asked until the user has tried to figure the question out for himself. As mentioned previously, people often learn faster when they have the opportunity to ask questions.

4.1.2 Typing and Editing Time

The test for time required to create and edit the letter was the same as the test used at the end of the Etude training session. A subject was given a letter to type and then given a marked up copy of the same letter. To complete the task, the content had to be letter perfect, and the outward appearance had to be reasonable. For Etude, this meant that mistakes in the editorial structure were tolerated if the outward appearance was still reasonable. For example, extra spacing between components was tolerated as long as the letter was still on one page. For the typewriter, this meant that margins did not have to be exact, and that corrections did not have to be as clean and well-aligned as they would be if the letter was actually mailed. The typewriter used in this study was an IBM Correcting Selectric II. This and all subsequent decisions about the typing and editing tasks were made carefully to avoid introducing experimental bias into the comparisons.

While a subject using Etude could make changes in the marked-up copy directly, what could the subject using a typewriter do? Since some of the changes were extensive (such as moving the last sentence in a paragraph to the start of that paragraph), the easiest way to make the changes was to retype the entire letter. While the actual operations in Etude and a typewriter were not the same, the functions were identical—the subject was to make a revised version of a letter that he had just typed. This emphasizes our interest in using functions (such as typing a letter) as the unit with which to measure speed of use, rather than measuring the time of individual tasks that make up the function.

Three sets of letters were used in each experiment—one at the end of the Etude training task, and one set each for the tasks measuring speed of use of Etude and a typewriter. This required different sets of letters to avoid practice effects resulting in greater speed from typing the same letter over and over again during the course of the experiment. The practice effect in retyping a letter to make the corrections was intentional, however.

Three letters of nearly equal length were used in the study, and the same set of editing tasks was applied to each one (though not necessarily in the same order in each letter):

1. Replace a character with another character.
2. Erase two words.
3. Move a sentence at the end of a paragraph to the start of that paragraph.
4. Split a paragraph into two smaller paragraphs.

These tasks were selected from Roberts' core learning experiment, in lieu of a standard set of typing and editing tasks to be used in such evaluations. Both the original and marked-up versions of each of the three letters are included in Appendix B.

The functions taught in the tutorial were not limited to those included in the experimental task, but included other functions that were considered to be basic to the task of typing letters even though they were not included here. Specifically, this included teaching subjects to type in formatted text where components were not already provided (e.g. if a phrase was to be italicized).

4.2 Experimental Protocol

Since each subject would be using both Etude and a typewriter, with several comparisons made between the two machines, it was important to control for as many extraneous variables as possible. One of the most prominent of these variables was the order in which the subject used the two machines. To control for this effect, two different experimental protocols were used; one in which Etude was used first, and the other in which the typewriter was used first. Half of the subjects were assigned to one protocol and half to another, with the assignments made at random. Figure 4-1 gives the experimental protocols for both orders of administration.

Typewriter first	**Etude first**
Introduction	Introduction
Typewriter tasks	Etude tutorial
Typewriter questionnaires	Etude practice tasks
Etude tutorial	Break
Etude practice tasks	Etude tasks
Break	Etude questionnaires
Etude tasks	Typewriter tasks
Etude questionnaires	Typewriter questionnaires
Conclusion	Conclusion

Figure 4-1: Experimental Protocols

Each subject typed three sets of letters during the course of the experiment. Even though the letters were the same length and included the same editing tasks, it was possible that the letters varied in difficulty. Since there were three letters, six possible ordering of letters were possible. Subjects were randomly assigned to one of the six orderings, with a nearly equal number of subjects assigned to each ordering.

Other variables include the time of day and day of the week on which an experiment was carried out. The experiments were conducted on Saturday mornings and afternoons, Sunday mornings and afternoons, and Monday, Tuesday, and Thursday evenings. These variables were recorded for each subject.

Throughout the experiment, care was taken to try to minimize the amount of anxiety induced by the experiment itself. For this reason, oral instructions were preferred to written instructions, even though written instructions assure greater uniformity of experimental treatment.

The most important aspect of the introduction was to have the subject read and sign a consent form, which gives a brief description of the experiment and a description of the rights of subjects. This form is included in Appendix B. Before the form was presented, the subjects were also given a brief oral explanation of the experiment and the nature of the consent form.

The MIT Committee for the Use of Humans as Experimental Subjects requires that a paragraph about medical care available to subjects be included in all consent forms. Since this was completely irrelevant to this experiment (barring the possibility of bizarre accidents occurring inside the building), it was separated from the rest of the form by a dotted line. To alleviate anxiety, subjects were assured that this part of the form was only a bureaucratic necessity; the top part of the form contained all the important information.

The principal task was to type and edit a letter (in the case of the typewriter, "editing" meant retyping). Subjects were instructed to work with speed and accuracy, as any typographic mistakes would have to be corrected. When using a typewriter, subjects set margins and tabs as they wished before beginning the task. When using Etude, the equivalent of inserting a plain piece of paper into the machine—using **retrieve document** to get the letter template—was done before beginning the task.

The starting time was recorded when the experimenter instructed the subject to start whenever ready. The finishing time was taken to be the time when the subject finished typing a perfect copy of the letter. Thus, proofreading time that resulted in correcting mistakes was included in the timing, but proofreading time that did not catch any mistakes was excluded.

If mistakes were caught after the typist had pulled the paper from the typewriter, he was instructed to correct the mistakes using the correcting feature of the typewriter, but not to worry about getting the alignment of the correction exactly right. This would compensate for typists who were not familiar with the alignment on this particular model typewriter.

The Etude program was started before the subject arrived. When he was ready to begin the tutorial, Etude was displaying the first page of the tutorial. The subject was told that the tutorial was about five pages long and would encourage him to try things out as he went along. He was also told that while the tutorial tried to be self-explanatory, he should feel free to ask questions if something unexpected happened.

In the conclusion, the subject was asked what he particularly liked and disliked about Etude. He was then given the opportunity to ask any questions he had about the experiment, Etude, or word processing systems in general. The subject's time slip from the temporary agency was filled out and the experiment completed.

4.3 Subject Selection

Twenty-five subjects were hired from two temporary agencies in the Boston area. The temporary agencies were told that the subjects should be office workers who did not have any word processing experience. They were not to be selected because of their inclination towards technical jobs or a technical environment. Each subject was paid for four hours work at a rate of five dollars an hour.

The number twenty-five was selected to allow for things to go wrong, since the time schedule was such that experiments could not be rescheduled. Rough simulation of the statistical tests showed that twenty subjects would be an adequate sample size. The margin of safety turned out to be important, since three subjects did not show up.

During the conclusion of one experiment, one subject revealed experience with a computer typesetting system. This data was discarded without further analysis,[5] leaving a sample size of twenty-one. All of the other subjects had indeed not had any text editing experience, with attitudes towards the technical environment ranging from enthusiastic to fearful.

4.4 Data Analysis

4.4.1 Recording Data

Data sheets were used to systematically record data. Figure 4-2 shows an empty data sheet for an "Etude first" experiment. A watch with a second hand was used to record times to the nearest five seconds. While Etude timings could have been made with the

[5] Since this revelation came at the end of the experiment, the training time had already been recorded. This subject had finished the training session ten minutes faster than anyone else had.

Introduction (sign consent form)

Tutorial:	Start tutorial	
	End tutorial	
	Start type	
	End type	
	Start edit	
	End edit	
Break		
Etude:	Start type	
	End type	
	Start edit	
	End edit	
Questionnaires:	STAI	
	SD	E
		P
		A
Break		
Typewriter:	Start type	
	End type	
	Start edit	
	End edit	
Questionnaires:	STAI	
	SD	E
		P
		A

Conclusion:	Likes	Dislikes

Remarks/Observations:	Good	Bad

Figure 4-2: Data Sheet for "Etude First" Experiment

computer, this method of measurement could not be extended to the typewriter. Seven measurements were derived from this data:

1. Training time, measured from the start of the tutorial to the end of the practice editing task. No time is subtracted for times spent when the tutorial was interrupted by machine failure (it *is* subtracted for the other time measurements).

2. Typing time, the difference between the finishing and starting times for the speed typing task.

3. Editing time, the difference between the finishing and starting times for the speed editing task.

4. STAI score, the sum of the twenty scales on the STAI form.

5. Evaluation score, the mean of the four Evaluation scales on the SD.

6. Potency score, the mean of the four Potency scales on the SD.

7. Activity score, the mean of the four Activity scales on the SD.

4.4.2 Hypothesis Testing

The independent variable in this experiment was the device with which the subjects typed and edited letters: Etude or the typewriter. The dependent variables were the seven measurements listed above, with the exception of training time (which was not compared). Each of these dependent variables had two sets of data associated with it, reflecting the two different treatments available with the independent variable. The training time measure had only one set of data associated with it. The order in which the treatments are presented reflects another variable which will be discussed below.

These measurements were used to judge the criteria in two different ways. In the case of training time, descriptive statistics were used because there is nothing firm from which to form a hypothesis. These statistics were derived the sample distribution function, and included an estimate of the 90th percentile. It is more interesting to get an estimate for the 90th percentile than to perform a hypothesis test based on an arbitrary value for this percentile. This leads to a statement that 90% of the subjects could learn to use Etude within a certain amount of time, but due to the small sample size it does not give a good indication of the accuracy of that estimate. On the other hand, reasonable confidence intervals for the median could be obtained with the given sample size [6, 73].

In most cases, the random variables are used to test hypotheses that are based on the criteria. The null hypothesis is that there is no difference between Etude and a typewriter for the given criterion. Four different alternative hypotheses are available:

1. Etude is better than a typewriter for the given criterion.
2. Etude is no worse than a typewriter for the given criterion.
3. Etude is no better than a typewriter for the given criterion.
4. Etude is worse than a typewriter for the given criterion.

A statistical test can either accept the null hypothesis or reject it in favor of one of the alternative hypotheses. Alternate hypotheses 1 and 3 above correspond to hypotheses for a one-tailed test, while hypotheses 2 and 4 correspond to a two-tailed test. These tests are performed in the same way, but a one-tailed test requires only half the significance level of a two-tailed test. A test for hypothesis 1 at the 0.02 significance level ($p < 0.02$) is the same as a test for hypothesis 2 with $p < 0.01$.

Accepting an alternate hypothesis is a much surer conclusion than accepting the null hypothesis. A test with $p < 0.05$ has only a 5% likelihood of rejecting a true null hypothesis in favor of an untrue alternate hypothesis. The other possible error is to accept an untrue null hypothesis instead of rejecting it for a true alternate hypothesis.

Hypothesis tests are designed in such a way that this type of error is much more likely to occur than the first type of error. Therefore, acceptance of the null hypothesis is always an uncertain result and should be treated very cautiously.

The following outline shows the various tests that Etude should satisfy in order to meet each particular ease of use criterion.

1. Criterion: Time to create and edit letters

    a. Hypothesis: Novices take no longer to type a letter using Etude than using a typewriter.

    b. Hypothesis: It is faster for novices to use Etude to create a revised letter than to use a typewriter.

2. Criterion: State anxiety

    a. Hypothesis: Users have no more anxiety when using Etude than when using a typewriter.

3. Criterion: User attitudes

    a. Hypothesis: Users have a favorable attitude towards Etude.

    b. Hypothesis: Users have at least as favorable an attitude towards Etude as they do towards a typewriter.

It should be made clear that the variable used in the hypotheses for user attitudes was the Evaluation score of the SD. This is the only score for which we can say that one end of the scale is favorable and the other end is unfavorable. Hypothesis tests were performed on the other attitude variables as a measure of attitudes, but the results do not bear directly on the ease of use criterion.

4.4.3 Nonparametric Statistical Tests

There are two different flavors of statistical tests which are available for experimental studies. One flavor assumes a parametric model of the underlying distributions of the random variable. In most cases, the random variable is assumed to have a normal underlying distribution. When this is the case, tests such as the Student *t*-test can be used.

There are different schools of thought on whether or not to use the normality assumption in the absence of compelling evidence either for or against the assumption. In some experimental designs, the parametric test may be more powerful than the nonparametric test. An experiment that assumes the parametric model would then be more likely to show a significant result than an experiment which did not assume that model.

In this study, all subjects received both treatments. This permitted the use of a matched-pairs test, which uses the differences between treatments for each subject as the basis for the test. This eliminates a major source of noise in the experiment, which is the difference between subjects reflected in their scores for a particular random variable. People vary greatly in their ability to perform complex cognitive tasks such as text processing and in their susceptibility to anxiety. By measuring the differences between treatments for each subject, the variance due to the differences between subjects is factored out. Variance is added due to the differing order of the treatments, but this can be handled by counterbalancing the experiment as was done here. Chapter 6 of Ledgard, Singer and Whiteside [57] contains a discussion of this issue and shows how it affected the outcome of one experiment considerably.

It turns out that assuming the nonparametric model in a matched-pairs test involves very little loss in power from the parametric model. The standard nonparametric test for this design, the Wilcoxon matched-pairs signed-rank test, is nearly as efficient as the standard parametric test, the S:udent *t*-test for matched pairs, in cases where the underlying distributions are indeed normal. The Wilcoxon test is usually better when the underlying distributions are not normal. In this case, the Wilcoxon test is the method of choice.

For learning time, typing time, editing time, and the STAI score, there was no particular indication that the normality assumption was wrong, so the above argument was important for determining which test to use. In the case of the SD, though, normality usually cannot be assumed [80]. SD scores range from a value of -3 to +3. When a small number of scales is being averaged to produce each score, as was the case in this study, values near the endpoints are not uncommon. Therefore, an assumption of the normal distribution cannot be justified. This left little choice but to use the Wilcoxon test for the comparison of SD scores.

The test of favorable attitudes differs from the other hypothesis tests in this study in that it does not involve a comparison with the typewriter. SD scores have a clear cutoff point at zero between opposing attitudes, so a hypothesis that a measure of central tendency is different from zero could readily be used. Usually, these tests are related to the process of establishing confidence intervals based on the mean, but these tests again assume normality. Confidence intervals based on the median can be constructed without this assumption [6, 73] and are used instead.

To summarize, training time was measured by a sample distribution function which can provide estimates for various percentiles, including a confidence interval for the median. User anxiety and time to create and edit letters were measured by comparing Etude to a typewriter. User attitudes were measured by testing for a non-zero median as well as by comparing Etude and the typewriter, using the Evaluation component of the SD.

Wilcoxon and Wilcox [121] givε a simple explanation of the use of the Wilcoxon test. Runyon [95] also gives an explanation, and includes a table for the test statistic *T* which is more conservative than the one used by Wilcoxon and Wilcox. Wilcoxon has described the reasoning behind the test [120]. Other statistics books can be consulted as well, e.g. Breiman [10, pp. 260–268].

4.5 Pre-Tests

Two pre-tests were run one week before the experiments were scheduled to begin, in order to correct problems occurring with Etude, the tutorial, or the experimental procedure. Several minor problems were caught in this fashion.

One problem reflected the ad-hoc nature of the construction of the typing and editing tasks. The original editing task contained six corrections instead of four. The additional tasks included inserting a sentence into a paragraph and erasing an arbitrary region. This made the task too big, in that too many corrections were spread over too little text. The extra tasks were eliminated and the letters revised to reflect these omissions.

The original typing task did not provide the subject with a notations component in the empty letter, requiring the subject to use **begin** rather than **go to**. This proved to be quite confusing for the subjects, since it occurred at the end of the letter. I decided it would be more useful to include a notations component in the standard letter.

The addition of backup facilities to Etude was made after the pre-test, when it became clear that the lack of system reliability required these precautions. The operation of the **back word** key was also changed, in response to the problems that the subjects had with it. Previously, **back word** had erased the space before the word in addition to the word itself, requiring the subject to type a space before retyping the word. This was changed, so that **back word** now erases the space after the word instead of the space before the word. This forced a few other changes in the operation of the command to ensure that the **back word** key could be used repeatedly to erase several words.

As mentioned in section 2.4.7, the tutorial was revised slightly after the pre-test. One chapter was omitted, another chapter was moved further back, and more examples were added toward the beginning.

Sometimes simple but important details can slip past the experimenter unnoticed until the pre-test reveals the flaw. In this experiment, the typewriter and the Nu machine were in different rooms. While the room with the Nu machine had a combination lock, the office with the typewriter had a standard lock. When I was locked out of that room on the evening of the second pre-test, it became apparent that I would need a key to that office in order to conduct the typewriter tests. While pre-tests may be used on a larger scale to produce a considerable refinement of a design, even small-scale usage of pre-tests is an extremely important part of designing an experiment.

4.6 Discussion

The major strength of this experimental design is that it can be used to evaluate an entire system, including interactions between all of the features. No sophisticated measuring equipment is required to produce a multi-dimensional ease of use evaluation of the system. The statistical tests used are quite powerful, easy to compute, and have a good likelihood of catching systematic effects with a sample size of twenty subjects.

The major weakness of the experimental design is that isolated features cannot be analyzed to determine their role in the results. Etude follows many ease of use guidelines, but this experiment cannot isolate a particular guideline to determine its usefulness.

This experimental design reflects the fact that this evaluation was intended as a way for computer scientists who have designed a particular system to get feedback on the success of their efforts. Unlike much of the experimental work in computer systems, it is not intended as a psychological investigation into the way people use computers. An experiment must be designed to accurately measure the goals of the experimenters in order to have internal validity.

Chapter Five

Results

5.1 Ease of Learning

Figure 5-1 shows the sample distribution function for training time. Table 5-1 contains statistics derived from this distribution. Statistics in this and all future tables were computed using the Consistent System on Multics [17].

Mean	Standard Deviation	Range	Median	95% Confidence Interval	90th Percentile
1:53:25	34:35	1:16:10 — 3:46:10	1:52:10	1:26:30 — 2:06:20	2:19:20

Table 5-1: Statistics for Training Time

The practical significance of this test is that it indicates that office workers who have never used a text processing system before *can* sit down at the machine and learn how to create and edit simple formatted documents, such as letters, within a half of a work day. Etude does appear to be easy to learn. With more careful attention paid to refinement of the tutorial, along the lines suggested by Al-Awar, Chapanis, and Ford [1], training time might be reduced even further.

5.2 Comparisons With the Typewriter

Table 5-2 summarizes the results of all the comparisons between Etude and the typewriter.

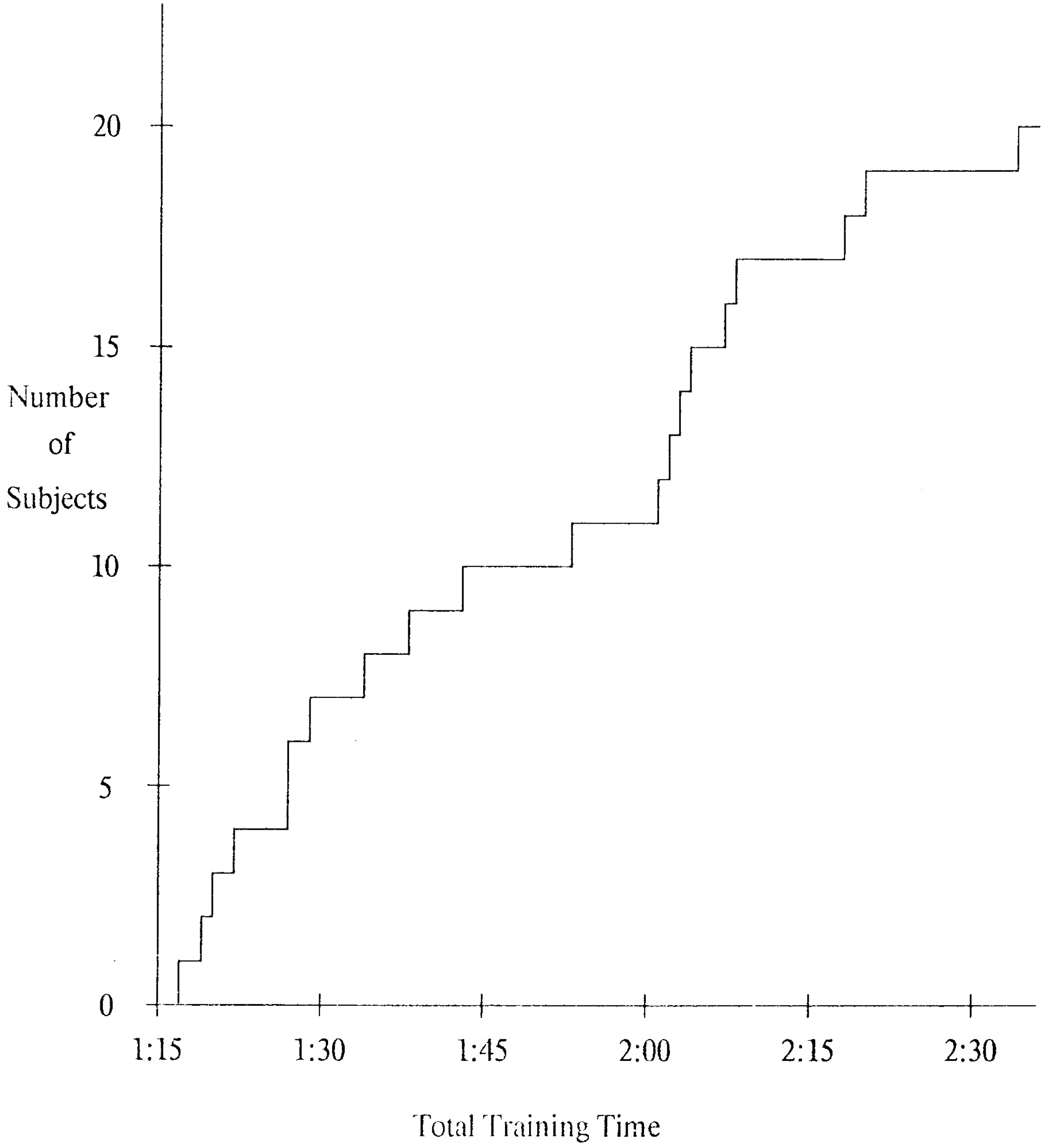


Figure 5-1: Sample Distribution Function for Training Time

Criterion	Etude	Typewriter	Significance
a) Ease of use once learned			
Mean typing time	16:35	6:55	0.00
Mean editing time	8:20	5:55	0.04
b) Anxiety factor			
Mean STAI score	40.95	41.33	0.96
c) User attitudes			
Mean evaluation score	1.12	0.69	0.08
Mean potency score	0.38	0.27	0.57
Mean activity score	–0.68	1.05	0.00

Table 5-2: Summary of Comparison Tests

5.2.1 Ease of Use Once Learned

Table 5-3 contains statistics for typing speed, while Table 5-4 contains statistics for editing speed. As shown above, Etude was slower than a typewriter for the typing task ($p < 0.01$) and no faster than a typewriter for the editing task ($p < 0.05$).

	Mean	Standard Deviation	Range	Median	95% Confidence Interval
Etude	16:35	5:40	9:15 — 29:55	15:05	13:05 — 20:45
Typewriter	6:55	2:00	3:40 — 10:25	6:20	5:45 — 7:45

Table 5-3: Statistics for Typing Task

	Mean	Standaı d Deviation	Range	Median	95% Confidence Interval
Etude	8:20	4:55	3:15 — 19:10	6:45	5:10 — 10:25
Typewriter	5:55	1:10	3:45 — 7:35	6:15	5:00 — 6:55

Table 5-4: Statistics for Editing Task

As mentioned before, the test of speed for novices is the least satisfactory test in the study. While it does measure ease of use for rank novices, that does not seem to be a particularly useful test. Better criteria would be the time it takes for an experienced but not necessarily expert user to complete these tasks or the amount of time it takes until a user can perform certain tasks faster than he can with a typewriter.

Even with these reservations, a working version of Etude would have to perform better on this test than the experimental version did. It appeared that Etude's poor response time was the main factor in its poor performance on this test. Since Etude could not keep up with the subjects' typing, they in turn could not correct their mistakes as they made them. Instead, they would have to use the editing commands to fix their mistakes. This is often less productive than correcting mistakes right away, and is especially true when the user is still learning to use the commands. Time was lost to go back and fix the mistakes, and sometimes more time was lost when the correction did not work as intended and the user had to try something else. The slow response to cursor movement contributed to the time lost in using the editing functions.

Again, it must be emphasized that this explanation is based on informal observation of users and engineering intuition. It is not based on any experimental data that can be extracted from the experiment. This experimental design was not intended to measure the effect of isolated features.

5.2.2 Anxiety Factor

Table 5-5 contains statistics derived from the STAI scores. There was no significant difference between Etude and the typewriter, so the null hypothesis cannot be rejected.

	Mean	Standard Deviation	Range	Median	95% Confidence Interval
Etude	40.95	8.12	25 — 54	41	36 — 47
Typewriter	41.33	9.60	23 — 64	43	35 — 45

Table 5-5: Statistics for STAI Scores

Figure 5-2 shows the differences in the means of each individual scale, showing which differences were shown to be significant in a Wilcoxon test. This presentation technique has been used in Semantic Differential research (Lucas' paper [61] is one example).

As mentioned in the previous chapter, acceptance of the null hypothesis must always be treated cautiously. There are two possible interpretations: one is that there really is no effect, and the other is that the noise in the experiment masked the effect. My own interpretation is that if there is a significant systematic effect, it is not substantial. Individual subjects showed highly significant differences, but these differences followed no pattern with respect to treatment differences. It seems to me that a systematic effect would have revealed more of a pattern.

Noise in the experiment was probably worse in the case of the anxiety factor than for the other criteria. As mentioned before, the experimental setting often induces anxiety in subjects, especially towards the beginning of the experiment. Subjects who used the typewriter first often received the first STAI less than twenty minutes after the

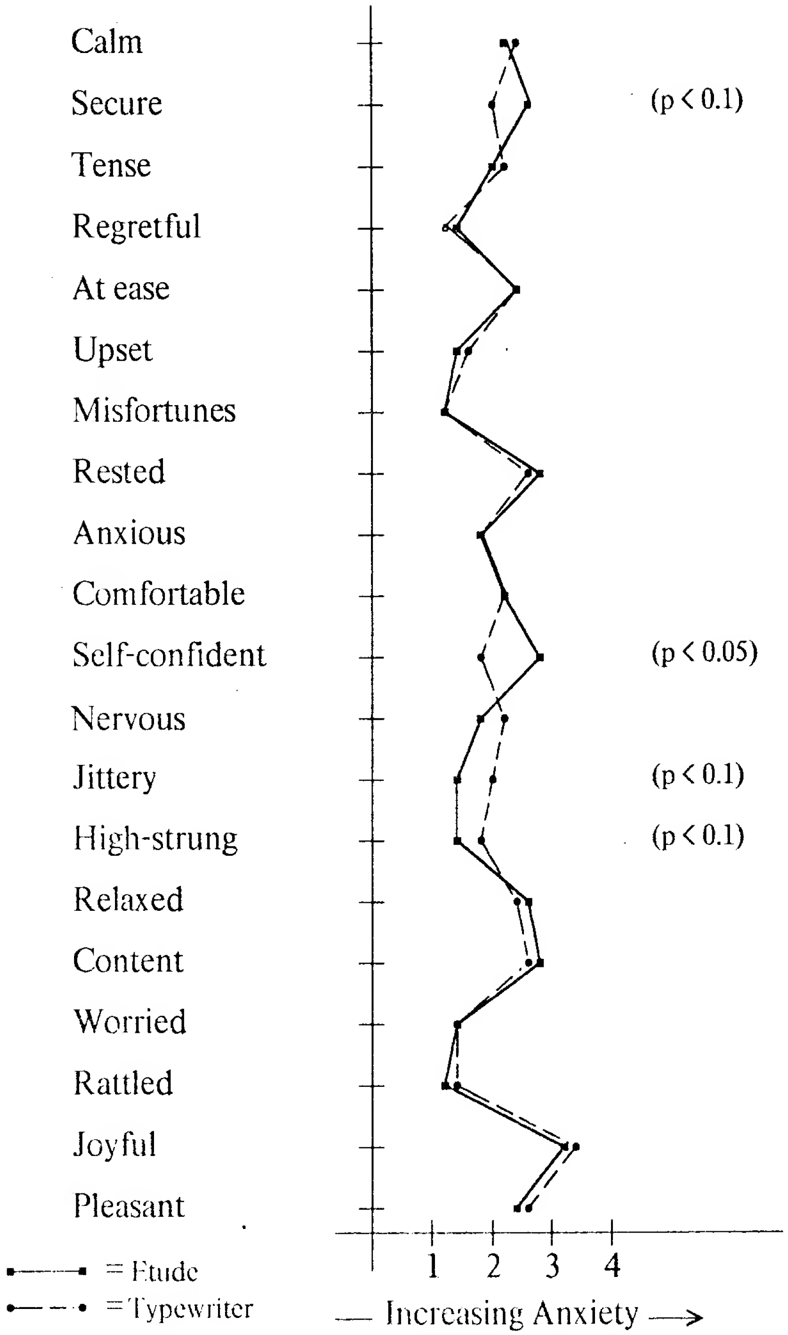


Figure 5-2: Differences in Individual STAI Scales

experiment started, whereas those who used Etude first often received the first STAI after more than an hour and a half. Another effect seemed to be that those subjects who were anxious while using Etude first had some of the anxiety carry over into the typing task. The interactions that are present might be beyond the reach of the simple analysis performed here. The interested reader might want to look at the data in Appendix C for more detailed information.

5.2.3 User Attitudes

Tables 5-6, 5-7, and 5-8 contain statistics derived from the Evaluation, Potency, and Activity scores of the SD. Evaluation scores for Etude were at least as high as those for the typewriter ($p < 0.05$), while Activity scores were higher for the typewriter ($p < 0.01$).

	Mean	Standard Deviation	Range	Median	95% Confidence Interval
Etude	1.12	1.12	-1.00 — 3.00	1.25	0.25 — 2.00
Typewriter	0.69	1.00	-1.75 — 2.50	0.50	0.25 — 1.25

Table 5-6: Statistics for Evaluation Scores

	Mean	Standard Deviation	Range	Median	95% Confidence Interval
Etude	0.38	0.53	-0.75 — 1.25	0.25	0.00 — 0.75
Typewriter	0.27	0.55	-1.00 — 1.75	0.25	0.00 — 0.50

Table 5-7: Statistics for Potency Scores

	Mean	Standard Deviation	Range	Median	95% Confidence Interval
Etude	-0.68	1.00	-2.00 — 0.50	-0.50	-1.25 — 0.00
Typewriter	1.05	0.63	-0.25 — 2.50	1.00	0.75 — 1.50

Table 5-8: Statistics for Activity Scores

Attitudes towards Etude were favorable even without the comparison to the typewriter, as the 95% confidence interval for Evaluation scores is completely on the positive pole of this dimension.

Figure 5-3 shows the differences in the means of each individual scale, with significant differences indicated. Graphs such have these have been referred to as "semantic profiles" [61].

While the two poles of the evaluation dimension can be categorized as measuring favorable or unfavorable attitudes, the same cannot be done with the potency and activity dimensions. Some people might welcome potent and/or active systems; others might be threatened by them. The favorability of one pole or another on these two dimensions varies with the individual.

On the basis of the evaluation scores, then, it is apparent that people enjoy using Etude. This was confirmed by remarks from the subjects during the closing interview. They also perceive Etude to be less active than the typewriter, in the SD sense of activity.

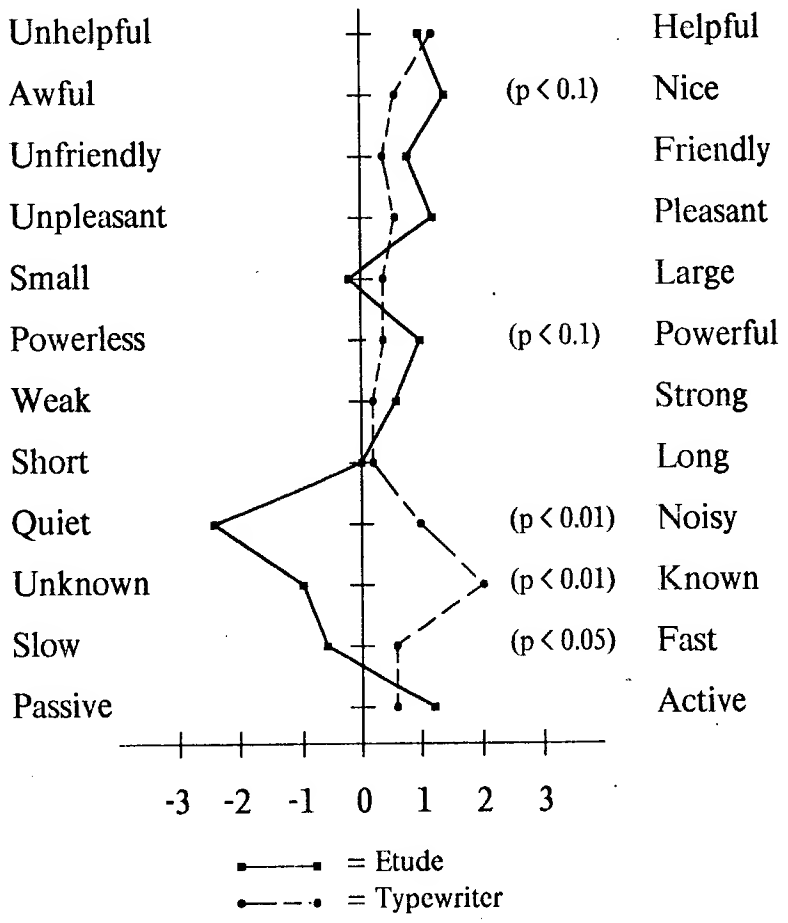


Figure 5-3: Differences in Individual SD Scales

5.3 Order Effects

Ten of the subjects used Etude and then the typewriter; the other eleven subjects used the devices in the other order. None of the measurements showed any significant effect from the order of the devices, as Table 5-9 indicates.

Criterion	First device	Second device	Significance
a) Ease of use			
Mean typing time	12:50	10:40	0.42
Mean editing time	7:35	6:40	0.51
b) Anxiety factor			
Mean STAI score	42.38	39.90	0.21
c) User attitudes			
Mean evaluation score	1.13	0.68	0.27
Mean potency score	0.34	0.68	0.88
Mean activity score	0.14	0.27	0.75

Table 5-9: Summary of Order Comparison Tests

5.4 Discussion

Etude generally satisfied three of the four criteria for ease of use. It was easy to learn, it evoked favorable attitudes, and it did not have a systematic effect on user anxiety. It was not easy for novices to use, however.

It is difficult to interpret the ease of learning statistic without more comparisons to work with. But it is useful to show that computer-naive people can begin to do useful work with a sophisticated text processing machine in less than half a working day. This compares favorably to many present text processing systems, which require a lenghty series of training sessions. Certainly, though, there is room for improvement.

The criterion of user attitudes is the least difficult to interpret. Subjects enjoyed using Etude, and also liked it at least as well as the typewriter they have used thoughout their working lives. This is truly a significant result, especially since the subject population included a large percentage of people who were not enthusiastic about computers and new office technology.

The anxiety criterion was not satisfied as strongly as would be desired. Given the state of most computer systems, though, the fact that Etude did not systematically induce anxiety in computer-naive subjects is an encouraging sign. Etude may not have conquered the problem of the anxiety factor, but it has had some success in the confrontation. The vagueness of the result points out the need for more research in measuring anxiety associated with the use of computers.

Although Etude was soundly defeated in speed comparisons with the typewriter, this was not a completely discouraging result. As was mentioned in chapter 3, the specific criterion was not particularly successful at measuring the general criterion. What we would like to know is how easy Etude is to use for people who have become comfortable using the system—that is, people who have progressed farther along the learning curve. Instead, we showed that Etude is not easy to use once you have just learned the system. While this is not a positive result, neither is it overwhelmingly negative. One must also take into account that many of the flaws that remained in Etude's design affected this criterion more strongly than the other criteria.

How much external validity does this experiment have? One generalization could probably be made without much trouble. In terms of the user population, the generalization from the subject population of computer-naive temporary office workers to a user population of computer-naive clerical workers seems fairly reasonable. Generalizing beyond this to either managerial workers or to people who are not computer-naive does not seem to be justifiable at this point.

Does the success of Etude in this experiment indicate that any system that follows the same user interface guidelines will also be easy to use? By itself, this experiment does no such thing. It is simply a small step toward providing experimental evidence to back up these guidelines. Repeated experiments involving systems following these guidelines are needed. Experiments that investigate individual guidelines and features are also needed, a point that is discussed further in the next chapter.

Chapter Six

Future Work

This results of this evaluation show that Etude has met most of its design goals regarding ease of use. It is easy to learn, it does not appear to have any systematic effect on user anxiety, and users have favorable attitudes towards it. Its main weakness is that it is slow to use, a problem which is being addressed in the new version of Etude.

There are many aspects of ease of use that this study did not cover:

1. Are there any isolated features that contribute significantly to Etude's ease of use?

2. How does Etude compare to other computer text processing systems?

3. How long does it take users to leave the novice stage?

4. How easy is Etude to use once users are no longer novices?

5. As Etude evolves into an integrated office workstation, will it remain easy to use?

The methods used in this study can be extended to attack these problems.

6.1 Analysis of Isolated Features

One way to examine the effect that isolated features have on Etude's overall ease of use is to construct two versions of the system, one that contains the feature and one that does not. Subjects could be taught to use one version of the system. After having practiced performing some tasks, they could be switched to the other version of the system and instructed to perform more tasks with that version. Half of the subject would learn one version first, half the other.

The State-Trait Anxiety Inventory and the Semantic Differential would be administered after the subject completed working with a particular version. These scores could be used in a matched-pairs test to test for changes in anxiety and attitudes, with a test for order effects as well. Speed of use could also be compared if the feature was hypothesized to have a strong effect in that area, but order effects based on practice might have a greater effect here. Learning rates could also be compared, but the between-subjects nature of the test would make it less powerful than the other tests.

Many features present themselves as likely candidates for experiments of this nature. Certainly a lot of work could be done with the **undo** key. It would be useful to test if the presence of even a single-step undo makes a significant contribution toward ease of use, especially in reducing the anxiety factor. Another test would involve comparing a single-step **undo** with a multiple-step **undo**. Yet another question would involve the effects of two **undo** operations in a row. In the current version of Etude, the second **undo** undoes the result of the first **undo**. Another alternative would be for the second **undo** to undo one operation further back in the session. Designers choosing among the various possibilities for implementing an **undo** facility currently do not have even informal guidelines to work with, much less any experimental data.

Other features which could be evaluated in this manner include:

- Different automatic completion facilities.

- A unified query-in-depth online assistance facility [30, 89].

- The use of confirmation.

6.2 Comparisons

While comparing Etude with a typewriter does have its positive aspects, comparisons with other computer text processing systems would be very interesting. It would be especially useful to compare Etude to another interactive editor and formatter. Xerox' Star [96], for example, shares some of the functional capabilities of Etude, but has a completely different style of interaction. The Star was also designed with ease of use as a primary consideration, further enhancing the interest in a comparison.

If subjects were taught to use both systems, the experiment would have to be designed so as not to overly penalize the first system taught to users. If naive subjects are used, this could be done by way of an introduction to the ideas behind computer text processing. This introduction would introduce basic ideas common to both systems, and would be timed separately from the training sequence for the individual systems. If more experienced subjects are used, the advantage for the first system might disappear altogether.

6.3 Long Term Evaluation

The process of learning to use Etude over a longer period of time could be the subject of a very valuable study. Performance, attitudes, and anxiety could be measured repeatedly for each subject. Such a repeated measures design would make it easy to use the same types of statistical tests used in this report. Comparisons between user performance at different points in time could show how long it takes for users to leave the novice stage and reveal the difference in the usage of various facilities (such as menus and abbreviations) between experienced and novice users. At the end of the study, expert user evaluations such as those made by Roberts [90] could be conducted. This type of study would take a long period of time and would require that Etude be capable of doing useful work at an acceptable rate of speed.

6.4 Enhancements to Etude

An evaluation similar to this one, but perhaps using different criteria, could be repeated with an enhanced version of Etude. This would be particularly useful in two areas:

- When Etude has been implemented so that it can run at a reasonable rate of speed and provides various facilities that were excluded in the prototype, it would be particularly interesting to check how Etude measures up to the criterion of ease of use once learned.

- When Etude is integrated with other facilities to form an integrated office workstation, an entire re-evaluation would be in order to see if the entire workstation is easy to use. Naturally, the experimental tasks would have to be changed and extended to reflect increased capabilities provided in areas such as office communication and database management.

As enhancements are made to Etude, experiments on the effect of these isolated features would be quite useful. When an integrated workstation is finished, a re-evaluation could be used to see if the workstation retains Etude's ease of use. It is one thing to be able to start to use a text processing system in less than two and a half hours. It would be quite another to be able to start to use an entire workstation in that amount of time.

Future experiments with Etude will require that the system run faster than the experimental version does. Efficiency was a major concern in designing the new version of Etude. This version is being implemented in MDL [31], a language developed by the MIT Laboratory for Computer Science's Programming Technology Group. It is similar to Lisp with the addition of data-type checking facilities. A machine-independent interpreter and compiler for MDL is being implemented in parallel with the new version of Etude. This will let Etude move at last to its intended environment—a powerful stand-alone computer with a bit-map display, such as the Apollo computer.

6.5 Conclusion

The amount of experimental work in the ease of use of computer systems reported in the literature has increased substantially in the past few years. System designers are beginning to get some hard data to back up the folk wisdom in the area of user interface design. Etude represents another step in the exploration of designing, building, and evaluating easy to use systems. This study is the Etude project's first contribution to the area of evaluation. The implementation of the new version of Etude will allow much more work to be done.

Appendix A

How to Use Etude

This tutorial has been used in an experimental study in which computer-naive subjects were taught to use Etude. The tutorial was typed and edited using Etude.

Subjects were presented with a running version of Etude that displayed a copy of the tutorial. Subjects practiced using Etude as they read their own copy of the tutorial, assured that they could not harm the original copy.

This version of the tutorial was produced by using a program that converts an Etude document into a Scribe [86] input file. This file was then edited to fit the style for a technical report.

1 Introduction

Etude is a machine that lets you type up written material, such as letters and reports, and see the material displayed in quality form. As you type in a document on the keyboard, Etude displays a copy of it on its video screen. Etude can make a paper copy of the document, but it will look no different than the copy you see on the screen.

You can also correct your work after it has been typed in, so changing things is no problem. Etude will show you the results of any changes that you make as you make them. So after you have typed a document, you can go back and correct typing mistakes, change the order of sentences, or do whatever other editing is needed. After you have the document looking just the way you want it, Etude can file it away in its memory for you to use again in the future.

One thing to remember about Etude is that you control it. It is your slave, but in order for you to order it around you must know what language it understands. The purpose of this tutorial is to teach you this language, which is very straight-forward. If you forget how to do something, Etude can remind you.

As parts of the language are introduced, practice using them. This copy of the tutorial is for your personal use. You can't hurt the original copy of a document when using Etude.

2 Moving Around in a Document

Notice the blinking arrow on the screen. This arrow is called a *cursor*. It is like a bookmark, in that it marks your place in a document. In order to make changes in a document on the screen the cursor (almost always) must point to the spot where the change is made. So you must move the cursor to where you want it.

One way to do this is by using the keys with arrows on them. If you want to move the cursor down, hit the key pointed down. Try this. The cursor always points at a letter or a space, so moving it up or down might cause it to move a little to the left or right as well. Move the cursor around in all directions, until you are comfortable doing it.

Now look at the bottom of the screen. This is not the end of the document, but no more will fit on the screen. The rest of the document is being stored by Etude. When you have finished reading the first page of the tutorial, how do you get to the second page? One way would be to keep on pressing the down arrow until you get to the last line of the page. Pressing it once more would then get you the next page.

It would be easier if you could just tell Etude to "go to the next page." Well, you can do this by pressing the key marked "go to," then the key marked "next," and finally the key marked "page." In this tutorial, we will print the names of special keys in bold, as in **go to next page**. Now to get back to this page, you would press **go to previous page**. Try going to the next page and coming back again.

3 Language Structure

Go to next page is typical of the instructions, or *commands*, that you give Etude to carry out. They are similar to commands in English, in that Etude understands some verbs, modifiers, and nouns. You give Etude commands by pressing the appropriate keys on the keyboard; verbs are on the left hand side, modifiers (including numbers) are in the middle, and nouns are on the right. You can do most things in Etude by putting a verb and one or two objects together to form a command.

Try moving around using commands such as **go to next paragraph**. You don't really need to hit **go to**; if you just use **next paragraph**, Etude is smart and knows you mean **go to**. Try using commands like **next paragraph**, **start of previous line**, **previous 10 words**, and **end of next 4 sentences**. Etude provides the "s" to make the nouns plural.

When you issue commands, Etude will display what it believes you have asked it to do on the fourth line from the top of the screen. This line is called the *response line*. The cursor will temporarily move to the response line until the command is actually carried out. As a precaution, Etude will often ask you to *confirm* a command before it goes ahead and does it. For example, if you pressed **erase paragraph**, Etude would highlight that paragraph by displaying white characters on a black background, and display "[confirm]" on the response line. If you press the **go ahead** key, Etude will display "[ok]" and proceed to carry out the command. If you change your mind and decide that this is the wrong area to erase, press the **cancel** key. This will keep the paragraph in the document, putting you back to where you started before hitting the **erase** key. Hit

erase line, notice the highlighting, and then hit **cancel**.

Sometimes Etude is slow to respond to commands. It tries to let you know what to expect by indicating the *response time* on the top line of the screen. If the response time is "very good," commands should be carried out very quickly. If the response time is "very poor," you may have to wait a while before you can see the results of the command. The other levels of response time are "good," "fair," and "poor." You can keep on going while Etude is working on a command, or you can wait for it to catch up if you need to see the result before going on.

4 The Help and Undo Keys

If you are ever unsure about what you can or should do next, you can press **help** to get an explanation of what you're doing. Hit **help** and observe what Etude tells you at the top of the screen. The next time that you hit a key, this information will go away.

Sometimes Etude is a little sloppy about removing **help** messages and other things from the screen. If you ever think that there is something being displayed that doesn't belong there, you can hit the **redisplay** key. This will clear the entire screen and then print the page over again. Try using **redisplay**.

If you have done something that you didn't intend to, like erase the wrong paragraph, you can correct it by pressing **undo**. This will reverse the effects of the last command you gave. For instance, if you have just erased a paragraph and want it back, you would just press **undo**. The paragraph would then return to the screen. You can only undo the last command that you gave, though.

Probably the easiest way to learn to use Etude is to give it a command, see what happens, and then immediately **undo** the command. You can do this as much as you like without changing the copy o: the document that you're working on.

For practice hit **erase next paragraph**, confirm it by pressing **go ahead**, and then **undo** it. Erase some different things now like **previous sentence**, **end of line**, **etc.**, and bring them back by pressing **undo** after each erasure.

5 How Documents Are Structured

When you're typing or editing a document, Etude must know if it is a report, a letter, or another type of document. Etude must also know what portion of a document you are working on. For instance, a letter may be divided into a return address, an address, a greeting, several paragraphs, a closing, and some notations. These different portions of a document are called *components*. Etude knows about many different types of documents and the different components that go with each document.

This tutorial is in the form of a report. The components of this document are labelled in the left hand margin. Etude tells you which components you are working on in the second line from the top of the screen. Try moving the cursor to this paragraph. Notice that this line displays "Components: Report/chapter 5/paragraph." This means that the cursor is inside a paragraph, the paragraph is inside chapter 5, and the chapter is inside the report.

Etude can display documents perfectly because for each component within a document, Etude knows what size to put the print in, what style type to use, what margins to use, and anything else it needs to know in order for it to look just right. For example, Etude knows that chapter titles in a report are printed in a bold type style, while the rest of the chapter is printed in a normal type style. The information in the top and left hand margins would not be printed on a paper copy of the document. It is displayed to help you while you are using Etude.

6 Inserting Text

The actual content of a document—the characters that are typed in—is often referred to as the *text* of the document. This distinguishes it from the *structure* of the document, which is indicated by the component names displayed in the margins. So the process of making additions to a document is referred to as "inserting text."

If you want to add some text to a document you simply move the cursor to the spot where you want the text inserted and type it. The text to the right of the cursor will move over to make room for the new material. In most cases, Etude will automatically break the lines for you so that they are all the same length. However, when you want to specify where a line ends (as in the return address of a letter), you can use the **new line** key like you would use the return key on a typewriter.

Sometimes Etude will not be able to keep up with your typing. In this case, Etude will just move the cursor instead of immediately displaying the new characters. Don't worry about this. If you need to see something that you have typed, just stop and let Etude catch up with you. The response time information on the top line of the screen will help you know what to expect.

Practice moving around and inserting text. You can erase the text either by using **undo** or by hitting the **back space** or **back word** keys enough times. **Back space** erases the character to the left of the cursor, and **back word** erases the entire word.

7 Component Names

You can use component names whenever Etude expects a noun in a command, as in **go to next chapter**. Since **chapter** does not appear on the keyboard, you would just type the word "chapter" after pressing **next**. Press **go ahead** to indicate that you are done typing the component name.

If you forget which component names are available at a particular spot, press **menu.** This will display a list of possible component names. To select one of the names from the menu, use the arrow keys to move the cursor to the name that you want. Etude tells you which menu item has been chosen by highlighting the item. You can then press **go ahead** to complete the selection or **cancel** to leave the menu. If you press **cancel,** you can then type the name of the item, just as if you hadn't used the menu. Press **next menu** and practice selecting items from the menu.

8 Regions

Most of Etudes editing commands work on "regions" of text. For instance when you say **erase next word,** the next word is the region of text that **erase** works on. **Previous 2 paragraphs, end of line,** and **chapter** all describe regions of text. But what happens if the region you want to work on is not a component, or something that has its own special key? Well, you can show Etude the region you want by using the **begin** and **end keys.**

Suppose you want to erase the middle of a sentence. First you would move the cursor to where you want to begin erasing. Then hit **erase begin.** Next move the cursor to the end of the area you want to erase and hit **end.** The region you have defined now goes from where the cursor was when you hit **begin** to where the cursor was when you hit **end.** After hitting **end,** the region will be highlighted, so you can be sure it is the one you want. Now to erase it hit **go ahead.** This confirms the command and erases the region. If you change your mind or Etude gave you the wrong region, hit **cancel** instead of **go ahead** and the whole command will be canceled.

Try erasing different regions of text in this document. Use the **begin** and **end** keys to define the regions. If you decide that you're working on the wrong region, hitting **begin** again will reset the region to start at your current position. You can also hit **begin** and move backwards through the document before hitting **end.** **Cancel** a few of the commands and be sure to bring back each erasure by using **undo.**

9 Moving Text

The **move** and **copy** commands are used to move text from one place to another. The only difference is that **copy** duplicates the region, so that it appears in both the old and new locations.

To use these commands, press **move** or **copy**, tell Etude what region you want moved, and then move the cursor to the place where you want the text to be. Hit **go ahead** to confirm the command.

To move a chapter to the end of the document, you would hit **move**, type "chapter" (ending with **go ahead**), press **end of document** to indicate the new location, and then press **go ahead.** Try moving a sentence to the end of a different paragraph.

10 Typing Components

To type formatted text, you have to tell Etude what component you want to use. You do this by using the **begin** and **end** keys. For example, to type the return address of a letter, you would do the following:

1. Press the **begin** key.
2. Type "return address" and hit **go ahead.**
3. Type in the text of the return address.
4. Press the **end** key.

To type the rest of the letter, you would type in the rest of the lette:'s components (the address, greeting, paragraphs, closing, etc.) the same way as you did the return address. Try adding a quotation to the middle of a paragraph.

Three components have their own special keys. You can press **paragraph**, **italic**, or **bold** instead of following step 2 above. You don't need to press **go ahead** after using one of the special keys.

A lot of times you will be using the same component over again. For instance, you probably will have lots of paragraphs in a row for some documents. You don't have to say **begin paragraph** and **end** for each new paragraph. After using **begin paragraph** to start the first paragraph, you can just hit **new component** to start each succeeding paragraph. This will end whatever component you were typing in and start a new component of the same type. After you have finished the last paragrpaph, press **end**. Try pressing **new component** at the end of a paragraph and typing some text.

New **component** can also be used to split components in two. Try pressing **new component** in the middle of a paragraph and see what happens; then **undo** it.

11 Filing, Retrieving, and Creating Documents

Etude can file a large number of documents for you. When you are finished with a document, press the **file document** key. Type the name that you wish to give to the document, and then hit **go ahead**. Etude will respond by telling you the exact location of the document in its filing system; you can ignore this. When you want to retrieve a document from the file, press the **retrieve document** key, type the document's name, and hit **go ahead**. It takes Etude more time to retrieve a document than to do most other commands.

Although you could create a new document completely from scratch, it is easier to use an "empty" document that contains no text, but does include the components that you will usually need for the particular document type. There is an empty letter filed under "letter." After retrieving the document from the file, you would say **go to return address** and start typing the return address. You would repeat this for each component in the letter. If the sample has some components that you don't need, move the cursor to the empty component and erase it. If you need something not included in the sample, add it using the **begin** and **end** keys. Use the **menu** key after pressing **begin** to see what components are available in a letter. They are different than those in a report.

To practice what you've learned in the tutorial, your instructor will give you a letter to type. Inform your instructor that you have reached the end of this tutorial.

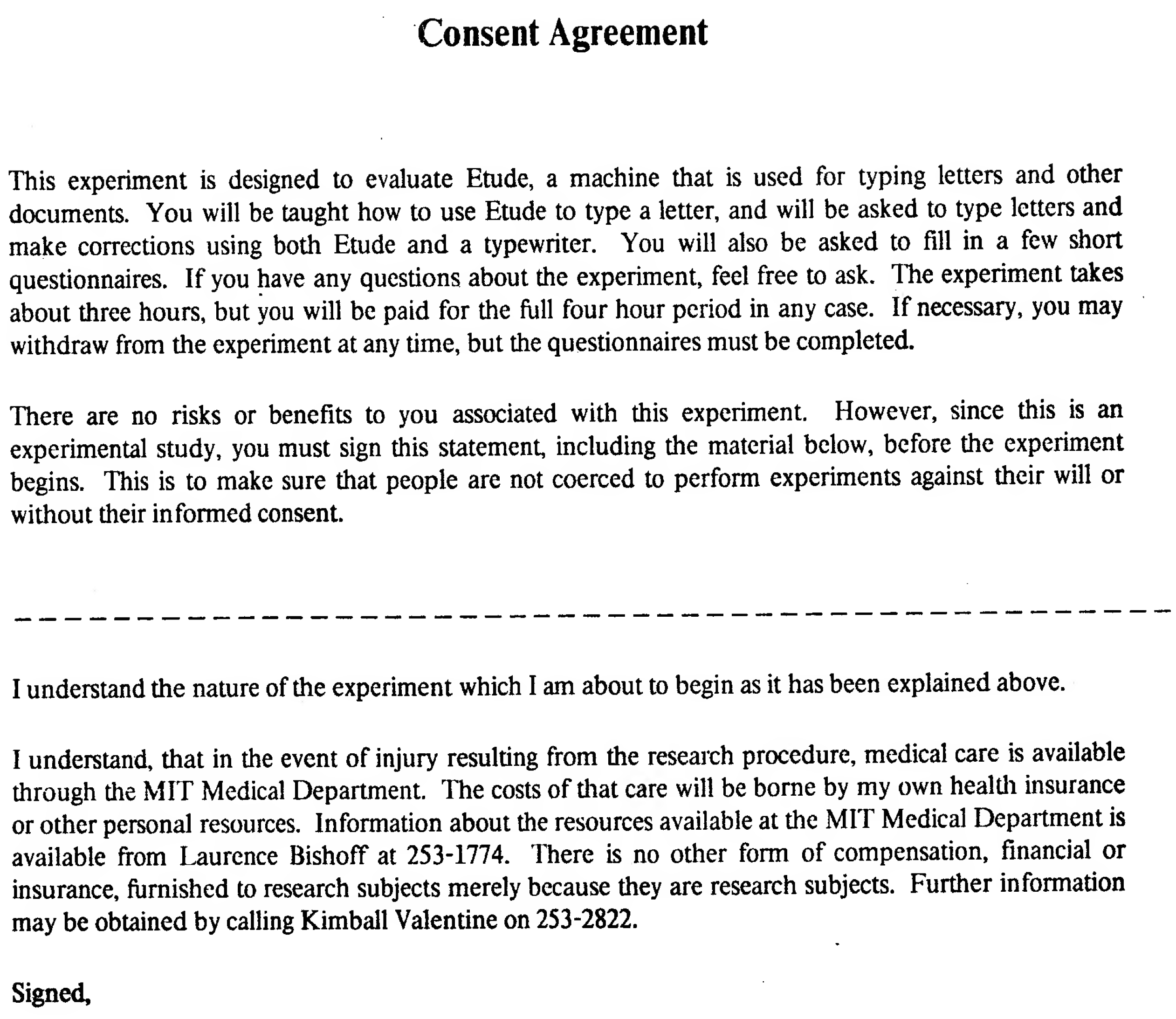
Appendix B

Experimental Materials

The following pages contain:

- A copy of the consent form that all subjects were required to sign.

- Copies of the letters that were used in the experiment.



Consent Agreement

This experiment is designed to evaluate Etude, a machine that is used for typing letters and other documents. You will be taught how to use Etude to type a letter, and will be asked to type letters and make corrections using both Etude and a typewriter. You will also be asked to fill in a few short questionnaires. If you have any questions about the experiment, feel free to ask. The experiment takes about three hours, but you will be paid for the full four hour period in any case. If necessary, you may withdraw from the experiment at any time, but the questionnaires must be completed.

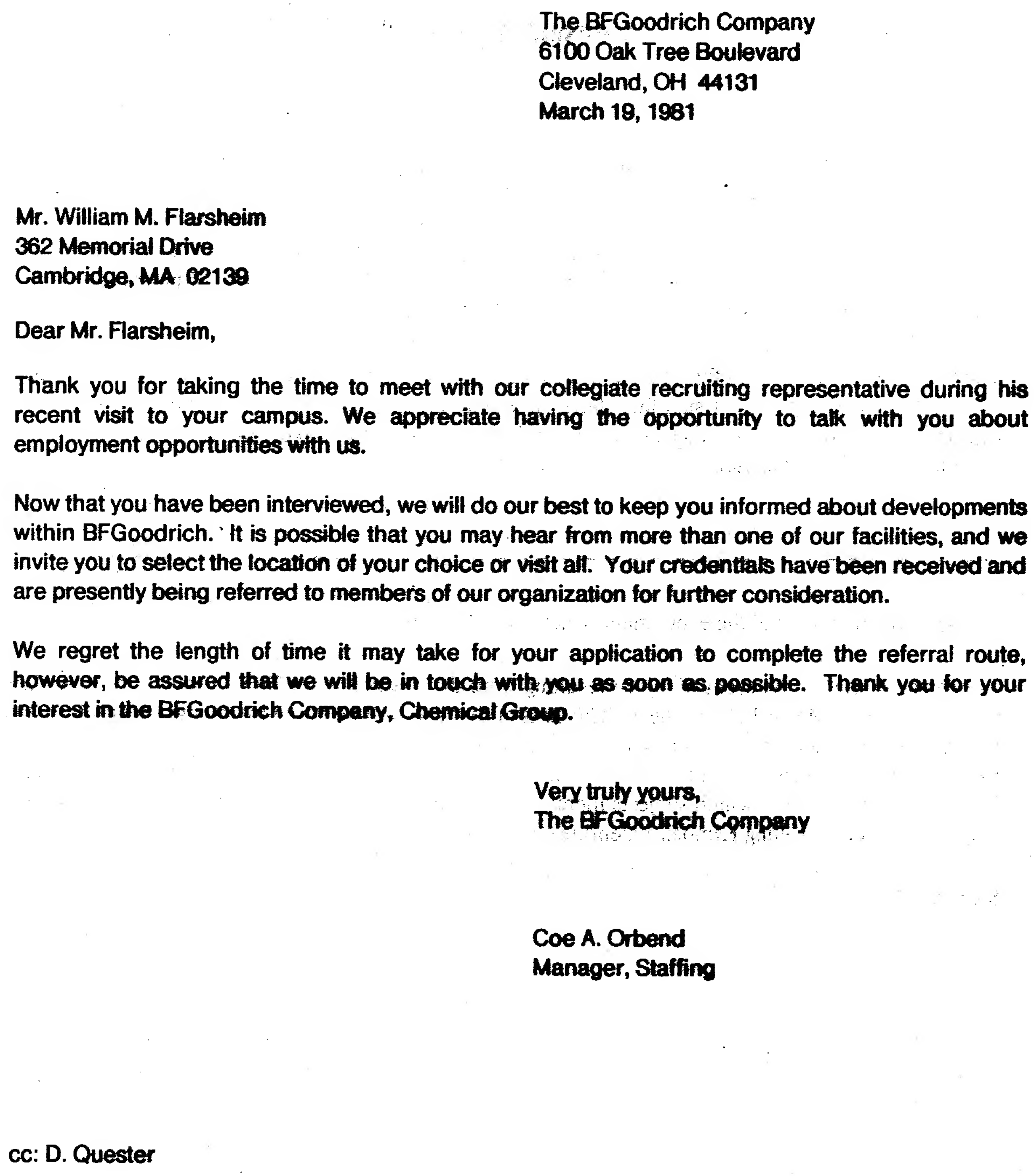
There are no risks or benefits to you associated with this experiment. However, since this is an experimental study, you must sign this statement, including the material below, before the experiment begins. This is to make sure that people are not coerced to perform experiments against their will or without their informed consent.

I understand the nature of the experiment which I am about to begin as it has been explained above.

I understand, that in the event of injury resulting from the research procedure, medical care is available through the MIT Medical Department. The costs of that care will be borne by my own health insurance or other personal resources. Information about the resources available at the MIT Medical Department is available from Laurence Bishoff at 253-1774. There is no other form of compensation, financial or insurance, furnished to research subjects merely because they are research subjects. Further information may be obtained by calling Kimball Valentine on 253-2822.

Signed,

Figure B-1: Consent Form



The BFGoodrich Company
6100 Oak Tree Boulevard
Cleveland, OH 44131
March 19, 1981

Mr. William M. Flarsheim
362 Memorial Drive
Cambridge, MA 02139

Dear Mr. Flarsheim,

Thank you for taking the time to meet with our collegiate recruiting representative during his recent visit to your campus. We appreciate having the opportunity to talk with you about employment opportunities with us.

Now that you have been interviewed, we will do our best to keep you informed about developments within BFGoodrich. It is possible that you may hear from more than one of our facilities, and we invite you to select the location of your choice or visit all. Your credentials have been received and are presently being referred to members of our organization for further consideration.

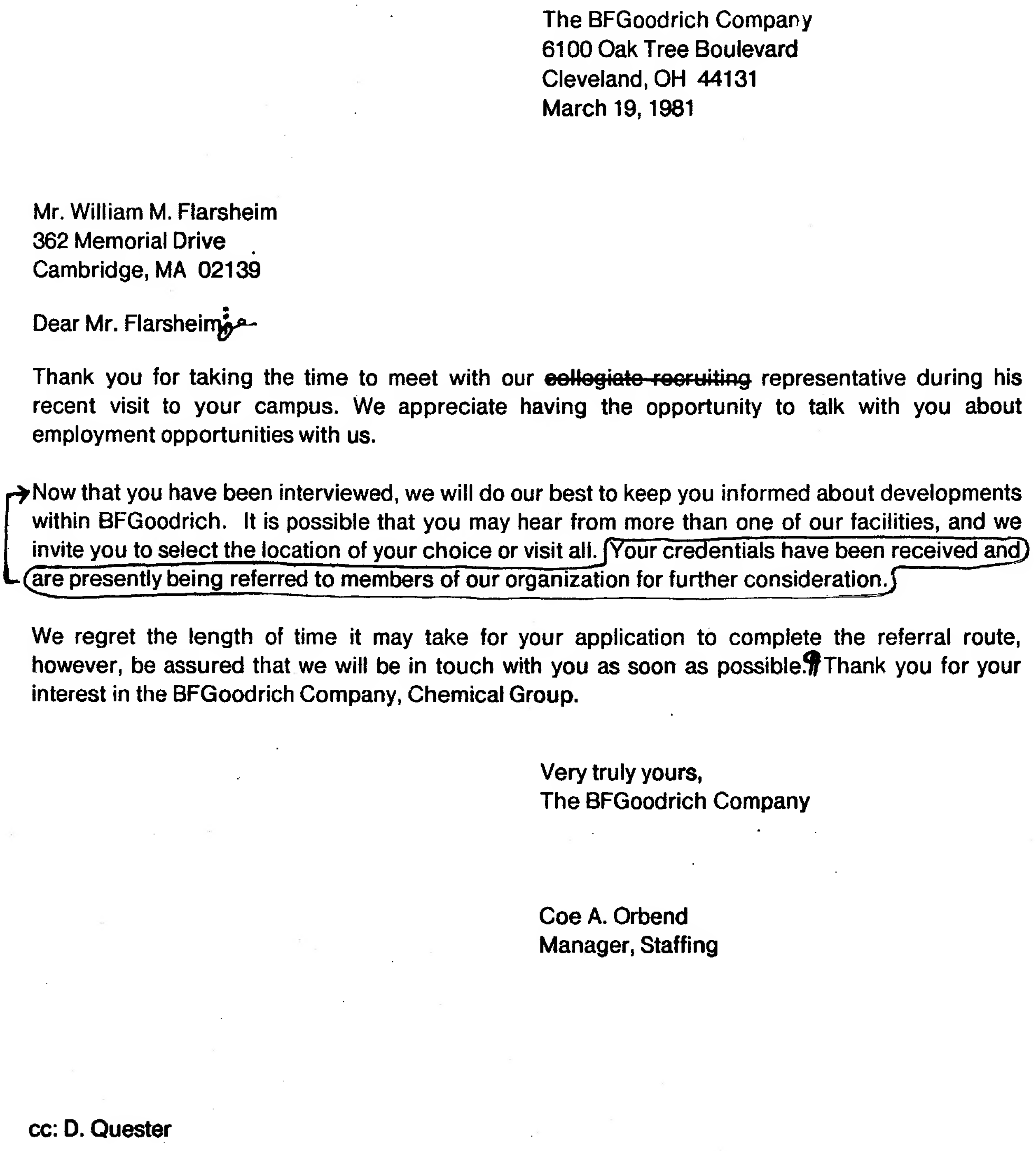
We regret the length of time it may take for your application to complete the referral route, however, be assured that we will be in touch with you as soon as possible. Thank you for your interest in the BFGoodrich Company, Chemical Group.

Very truly yours,
The BFGoodrich Company

Coe A. Orbend
Manager, Staffing

cc: D. Quester

Figure B-2: "Job" Letter—Original Copy



The BFGoodrich Company
6100 Oak Tree Boulevard
Cleveland, OH 44131
March 19, 1981

Mr. William M. Flarsheim
362 Memorial Drive
Cambridge, MA 02139

Dear Mr. Flarsheim:

Thank you for taking the time to meet with our ~~collegiate recruiting~~ representative during his recent visit to your campus. We appreciate having the opportunity to talk with you about employment opportunities with us.

Now that you have been interviewed, we will do our best to keep you informed about developments within BFGoodrich. It is possible that you may hear from more than one of our facilities, and we invite you to select the location of your choice or visit all. Your credentials have been received and are presently being referred to members of our organization for further consideration.

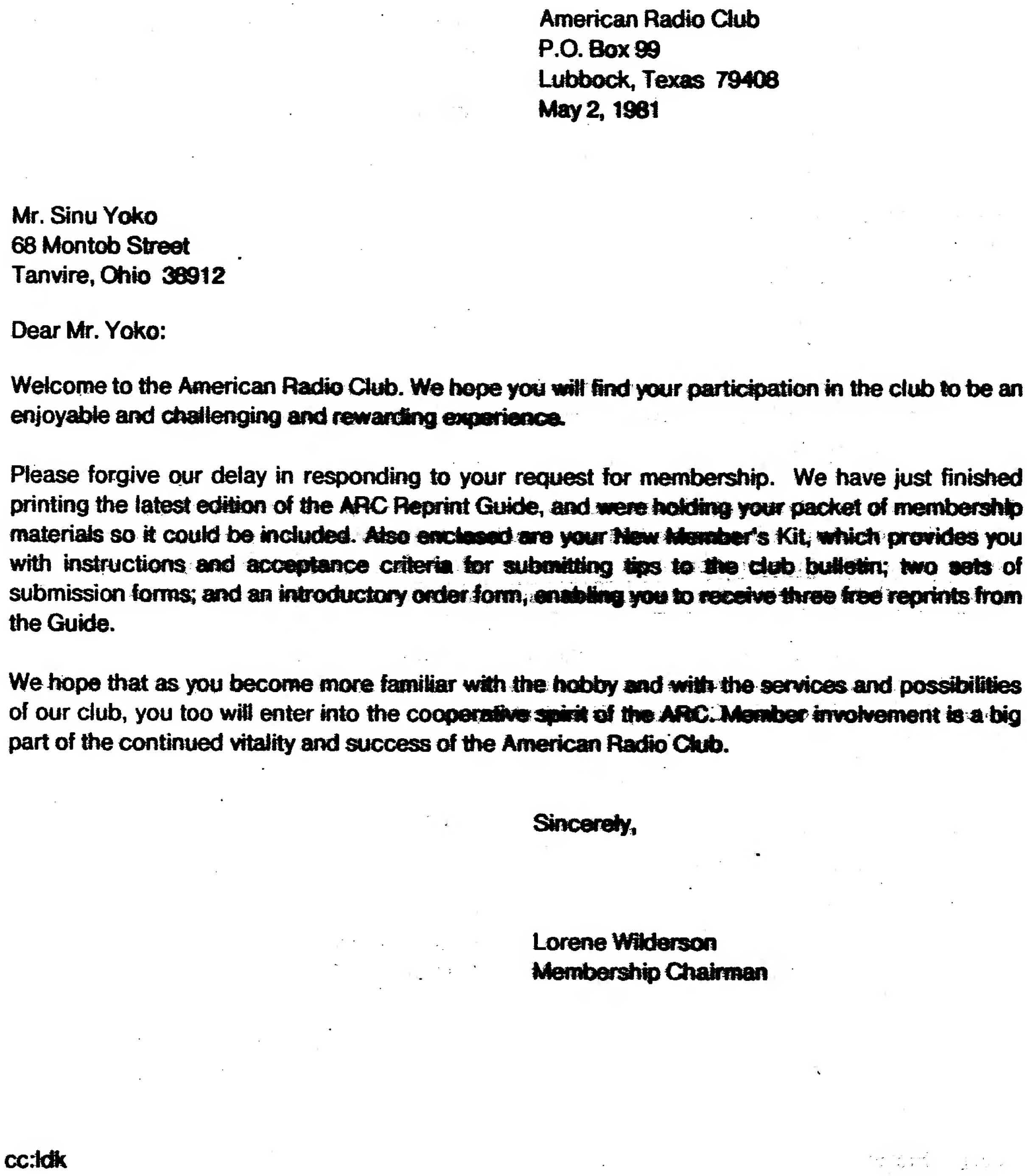
We regret the length of time it may take for your application to complete the referral route, however, be assured that we will be in touch with you as soon as possible. ¶ Thank you for your interest in the BFGoodrich Company, Chemical Group.

Very truly yours,
The BFGoodrich Company

Coe A. Orbend
Manager, Staffing

cc: D. Quester

Figure B-3: "Job" Letter—Marked Up Copy



American Radio Club
P.O. Box 99
Lubbock, Texas 79408
May 2, 1981

Mr. Sinu Yoko
68 Montob Street
Tanvire, Ohio 38912

Dear Mr. Yoko:

Welcome to the American Radio Club. We hope you will find your participation in the club to be an enjoyable and challenging and rewarding experience.

Please forgive our delay in responding to your request for membership. We have just finished printing the latest edition of the ARC Reprint Guide, and were holding your packet of membership materials so it could be included. Also enclosed are your New Member's Kit, which provides you with instructions and acceptance criteria for submitting tips to the club bulletin; two sets of submission forms; and an introductory order form, enabling you to receive three free reprints from the Guide.

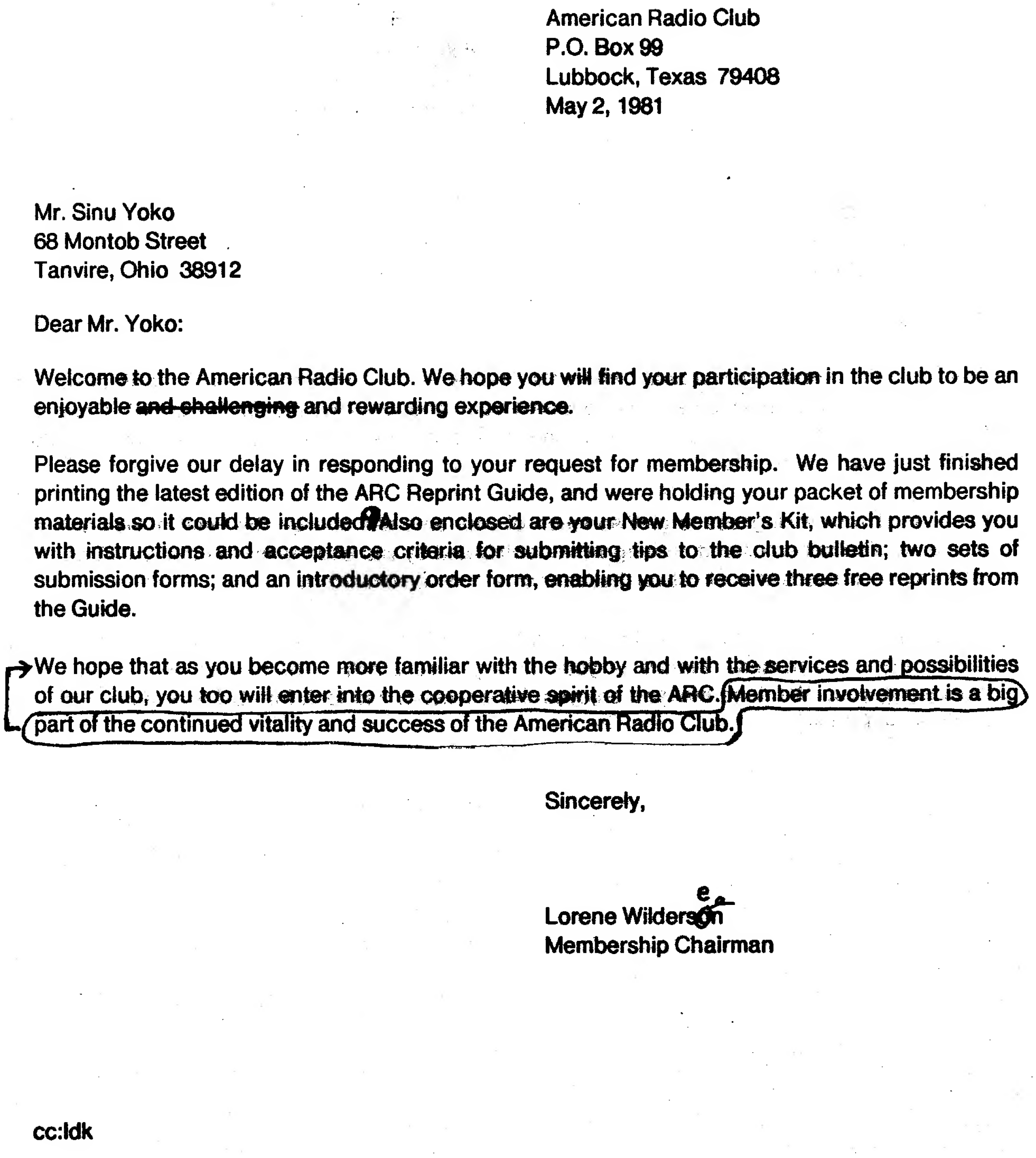
We hope that as you become more familiar with the hobby and with the services and possibilities of our club, you too will enter into the cooperative spirit of the ARC. Member involvement is a big part of the continued vitality and success of the American Radio Club.

Sincerely,

Lorene Wilderson
Membership Chairman

cc:ldk

Figure B-4: "Radio" Letter—Original Copy



American Radio Club
P.O. Box 99
Lubbock, Texas 79408
May 2, 1981

Mr. Sinu Yoko
68 Montob Street
Tanvire, Ohio 38912

Dear Mr. Yoko:

Welcome to the American Radio Club. We hope you will find your participation in the club to be an enjoyable ~~and challenging~~ and rewarding experience.

Please forgive our delay in responding to your request for membership. We have just finished printing the latest edition of the ARC Reprint Guide, and were holding your packet of membership materials so it could be included. ¶Also enclosed are your New Member's Kit, which provides you with instructions and acceptance criteria for submitting tips to the club bulletin; two sets of submission forms; and an introductory order form, enabling you to receive three free reprints from the Guide.

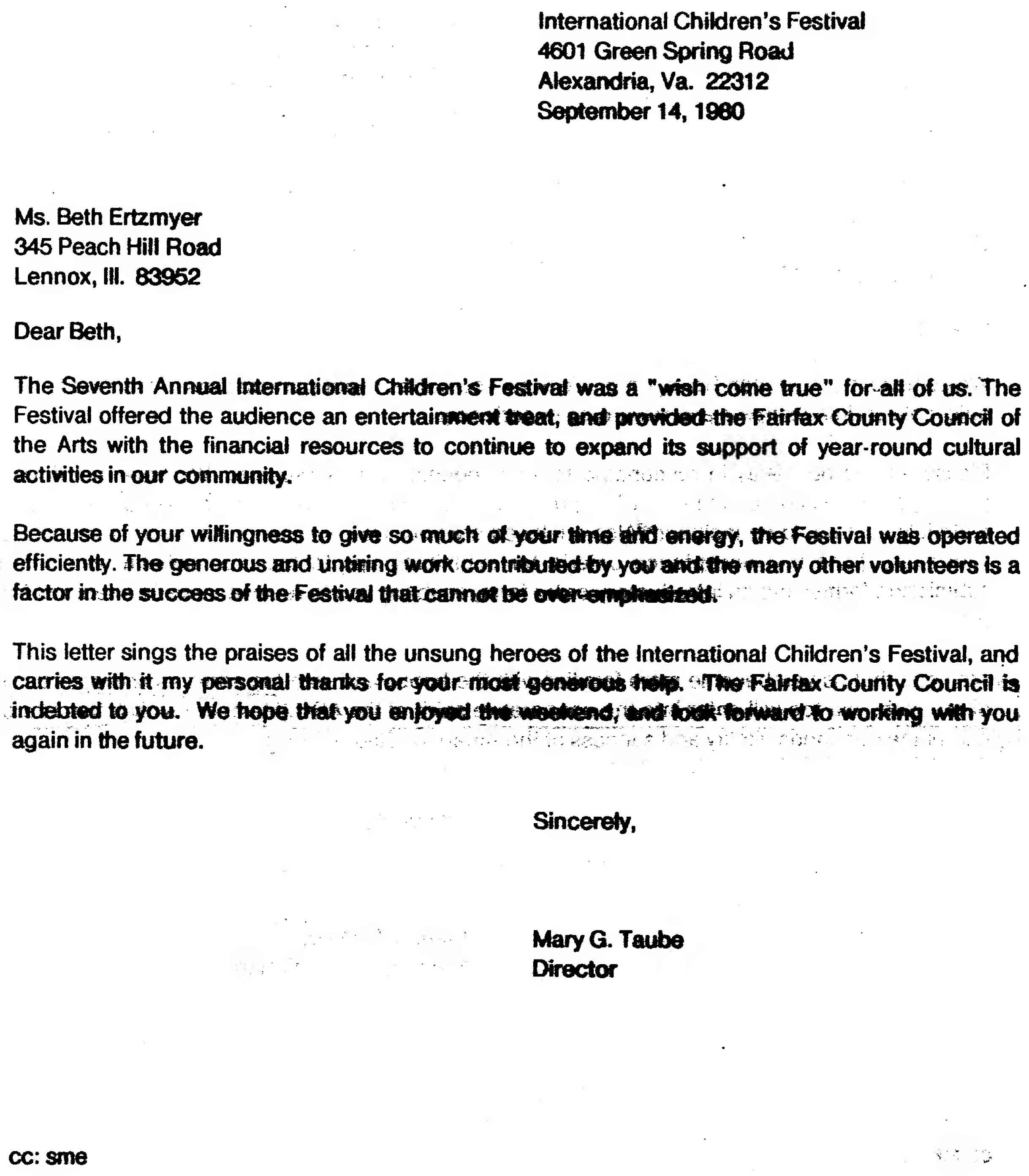
We hope that as you become more familiar with the hobby and with the services and possibilities of our club, you too will enter into the cooperative spirit of the ARC. Member involvement is a big part of the continued vitality and success of the American Radio Club.

Sincerely,

Lorene Wilders~~o~~en
Membership Chairman

cc:ldk

Figure B-5: "Radio" Letter—Marked Up Copy



International Children's Festival
4601 Green Spring Road
Alexandria, Va. 22312
September 14, 1980

Ms. Beth Ertzmyer
345 Peach Hill Road
Lennox, Ill. 83952

Dear Beth,

The Seventh Annual International Children's Festival was a "wish come true" for all of us. The Festival offered the audience an entertainment treat, and provided the Fairfax County Council of the Arts with the financial resources to continue to expand its support of year-round cultural activities in our community.

Because of your willingness to give so much of your time and energy, the Festival was operated efficiently. The generous and untiring work contributed by you and the many other volunteers is a factor in the success of the Festival that cannot be over-emphasized.

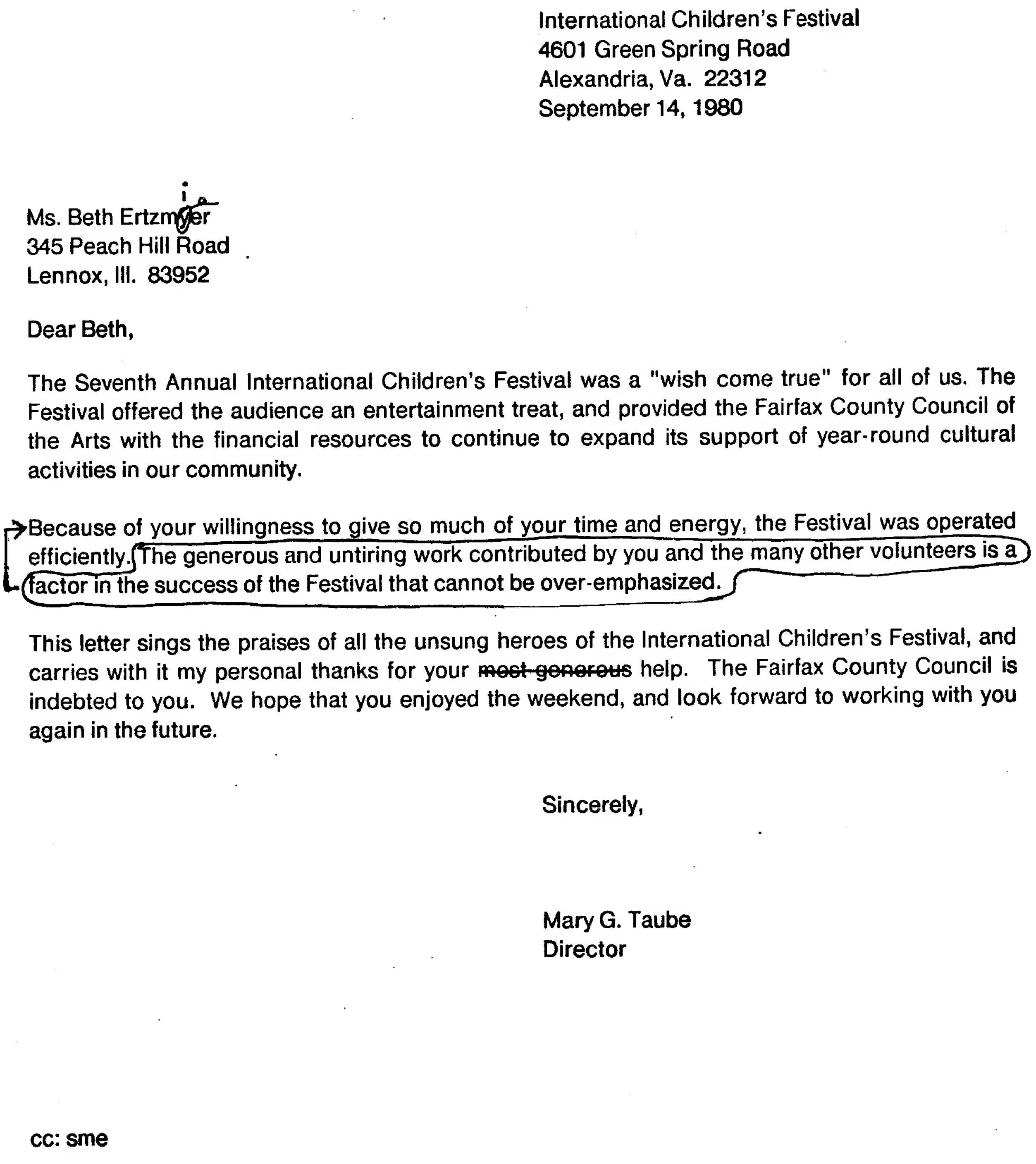
This letter sings the praises of all the unsung heroes of the International Children's Festival, and carries with it my personal thanks for your most generous help. The Fairfax County Council is indebted to you. We hope that you enjoyed the weekend, and look forward to working with you again in the future.

Sincerely,

Mary G. Taube
Director

cc: sme

Figure B-6: "Children" Letter—Original Copy



International Children's Festival
4601 Green Spring Road
Alexandria, Va. 22312
September 14, 1980

Ms. Beth Ertzmyer
345 Peach Hill Road
Lennox, Ill. 83952

Dear Beth,

The Seventh Annual International Children's Festival was a "wish come true" for all of us. The Festival offered the audience an entertainment treat, and provided the Fairfax County Council of the Arts with the financial resources to continue to expand its support of year-round cultural activities in our community.

Because of your willingness to give so much of your time and energy, the Festival was operated efficiently. The generous and untiring work contributed by you and the many other volunteers is a factor in the success of the Festival that cannot be over-emphasized.

This letter sings the praises of all the unsung heroes of the International Children's Festival, and carries with it my personal thanks for your ~~most generous~~ help. The Fairfax County Council is indebted to you. We hope that you enjoyed the weekend, and look forward to working with you again in the future.

Sincerely,

Mary G. Taube
Director

cc: sme

Figure B-7: "Children" Letter—Marked Up Copy

Appendix C

Data

These tables represent all the data recorded on the experiment datasheets. Each table is displayed on a subject-by-subject basis. The number in columns marked *S* refer to a unique subject number. Subject number 1 in one table is the same subject number 1 in any other table, and so on for each of the twenty-one subjects.

		Typing task		Editing task	
S	Training Task	Etude	Typewriter	Etude	Typewriter
1	1:18:30	13:50	6:05	5:15	5:30
2	2:33:35	9:20	3:40	16:05	5:15
3	2:00:55	24:10	9:20	3:35	7:20
4	1:42:25	22:50	6:40	4:30	7:15
5	2:06:20	23:45	6:20	5:40	5:00
6	1:33:35	11:30	10:25	3:55	6:25
7	2:03:25	17:10	4:15	19:10	6:40
8	1:52:20	15:05	9:40	7:50	7:20
9	1:26:30	20:55	5:15	9:15	7:35
10	1:26:20	10:20	5:25	3:40	3:45
11	3:46:10	9:30	4:25	8:20	5:00
12	1:37:30	20:45	6:15	16:55	6:55
13	2:17:40	29:55	7:45	16:20	6:20
14	1:19:05	14:35	7:10	5:55	6:25
15	2:02:55	13:45	10:15	3:15	5:00
16	1:28:10	9:15	6:20	11:00	7:05
17	2:07:50	13:45	5:45	10:25	5:40
18	1:21:25	18:25	10:05	6:45	6:15
19	2:19:20	20:45	6:20	7:00	5:05
20	1:16:10	13:05	6:45	5:30	4:05
21	2:01:10	15:15	6:15	5:10	4:00

Table C-1: Time Measurements

	STAI		Evaluation		Potency		Activity	
S	Etude	Typewriter	Etude	Typewriter	Etude	Typewriter	Etude	Typewriter
1	32	32	1.25	0.50	0.25	0.75	–0.75	0.75
2	54	56	2.00	1.50	0.50	0.75	–1.75	0.75
3	34	26	3.00	–1.75	0.75	–1.00	0.50	–0.25
4	45	44	1.25	0.50	–0.50	0.50	–1.50	1.00
5	47	44	1.25	1.25	0.00	0.00	–0.50	0.25
6	31	45	0.50	0.50	0.00	0.25	–0.50	1.75
7	46	45	0.50	0.75	0.50	0.00	–1.00	1.75
8	28	64	2.50	2.25	0.50	0.50	0.00	1.00
9	41	39	0.75	0.50	0.25	0.50	–0.75	1.75
10	51	23	1.50	0.00	–0.75	–0.50	0.00	1.00
11	51	41	–1.00	0.25	0.00	0.25	–1.25	1.50
12	41	41	–0.50	–0.75	0.00	0.00	–2.00	0.50
13	51	53	0.25	0.25	1.00	1.00	–0.50	0.75
14	36	40	2.00	1.50	0.25	0.00	0.00	1.00
15	38	46	0.25	1.00	0.25	0.25	–1.00	0.50
16	38	44	1.75	1.00	0.00	0.25	0.00	1.50
17	44	44	1.25	0.75	1.25	0.00	–0.50	0.75
18	37	32	0.25	0.00	1.00	0.25	–1.25	1.25
19	48	43	2.50	–0.25	0.75	0.00	–1.50	1.50
20	25	35	2.75	2.50	0.75	0.25	0.25	0.50
21	42	31	–0.50	2.25	1.25	1.75	–0.25	2.50

Table C-2: Questionnaire Measurements

S	First Device	Order of Letters	Day	Time	Number of Failures	Response Time
1	Etude	JRC	Saturday	Morning	1	2
2	Etude	RCJ	Saturday	Afternoon	6	1
3	Etude	CJR	Sunday	Morning	2	1
4	Etude	RCJ	Sunday	Afternoon	1	1
5	Typewriter	RJC	Monday	Evening	1	1
6	Typewriter	RCJ	Tuesday	Evening	4	2
7	Typewriter	CRJ	Thursday	Evening	3	1
8	Typewriter	RJC	Saturday	Morning	1	1
9	Etude	RJC	Saturday	Afternoon	1	1
10	Typewriter	JCR	Monday	Evening	2	1
11	Etude	JRC	Tuesday	Evening	5	2
12	Etude	CJR	Thursday	Evening	1	2
13	Etude	RCJ	Saturday	Morning	0	1
14	Typewriter	RJC	Saturday	Afternoon	1	1
15	Typewriter	JRC	Sunday	Morning	1	1
16	Typewriter	CRJ	Sunday	Afternoon	0	1
17	Etude	CJR	Monday	Evening	4	3
18	Typewriter	JCR	Tuesday	Evening	0	1
19	Etude	JRC	Thursday	Evening	3	3
20	Typewriter	CRJ	Saturday	Morning	0	1
21	Typewriter	JCR	Sunday	Afternoon	2	1

Order of Letters:

J: “Job” letter

R: “Radio” letter

C: “Children” letter

Response Time:

1: Best

3: Acceptable

5: Worst

Table C-3: Miscellaneous Data

	Evaluation				Potency				Activity			
S	1	2	3	4	5	6	7	8	9	10	11	12
1	2	2	0	1	0	0	2	–1	–3	0	–2	2
2	1	2	2	3	–3	3	2	0	–2	–3	–3	1
3	3	3	3	3	–2	3	2	0	–3	3	–1	3
4	2	2	0	1	–2	1	–1	0	–3	–3	–2	2
5	1	2	0	2	0	0	0	0	–3	0	–1	2
6	–2	1	1	2	0	0	0	0	–3	–1	1	1
7	2	0	0	0	2	0	0	0	0	–3	–1	0
8	3	2	2	3	0	2	0	0	–3	0	2	1
9	2	0	0	1	0	1	0	0	1	–2	–2	0
10	1	2	1	2	–2	1	0	–2	–2	–1	2	1
11	–1	–1	0	–2	0	0	0	0	–2	–2	–1	0
12	1	–1	–1	–1	0	1	0	–1	–3	–3	–3	1
13	0	0	1	0	–1	2	1	2	–2	1	–2	1
14	3	2	1	2	0	0	1	0	–3	1	2	0
15	–2	2	0	1	0	1	0	0	–3	–1	–1	1
16	2	2	1	2	0	0	0	0	–2	–2	2	2
17	1	1	2	1	1	1	2	1	–2	–1	–1	2
18	–1	1	0	1	0	2	0	2	–3	–2	–2	2
19	2	3	2	3	0	3	0	0	–2	–3	–1	0
20	3	3	2	3	2	1	1	–1	–3	2	–1	3
21	–2	1	0	–1	3	1	1	0	–3	0	1	1

1: helpful — unhelpful
2: nice — awful
3: friendly — unfriendly
4: pleasant — unpleasant

5: large — small
6: powerful — powerless
7: strong — weak
8: long — short

9: noisy — quiet
10: known — unknown
11: fast — slow
12: active — passive

Table C-4: Individual SD Scale Scores for Etude

	Evaluation				Potency				Activity			
S	1	2	3	4	5	6	7	8	9	10	11	12
1	1	0	0	1	1	2	0	0	2	2	0	–1
2	–3	3	3	3	0	0	0	3	3	3	–1	–2
3	–1	–3	0	–3	0	–3	–1	0	2	2	–3	–2
4	0	1	0	1	0	1	1	0	2	3	1	–2
5	2	2	2	–1	0	0	0	0	–2	2	1	0
6	2	0	0	0	0	1	0	0	1	3	2	1
7	3	0	0	0	0	0	0	0	0	3	2	2
8	3	2	2	2	2	0	0	0	1	0	2	1
9	2	0	0	0	0	1	0	1	2	3	1	1
10	0	0	0	0	–1	–1	0	0	1	3	–1	1
11	2	0	0	–1	1	0	0	0	2	2	1	1
12	–1	–2	–1	1	1	0	–1	0	2	1	–1	0
13	1	0	0	0	1	1	2	0	2	–2	2	1
14	2	2	0	2	0	0	0	0	–1	2	2	1
15	2	2	0	0	0	1	0	0	0	0	2	0
16	2	1	0	1	1	0	0	0	1	2	1	2
17	1	2	–1	1	1	–1	–1	1	1	2	–1	1
18	–1	0	0	1	0	1	0	0	1	2	0	2
19	1	–1	1	–2	0	1	–1	0	2	3	1	0
20	3	2	3	2	–1	1	2	–1	–2	3	–1	2
21	3	3	0	3	1	3	3	0	2	2	3	3

1: helpful — unhelpful
2: nice — awful
3: friendly — unfriendly
4: plcasant — unpleasant

5: large — small
6: powerful — powerless
7: strong — weak
8: long — short

9: noisy — quiet
10: known — unknown
11: fast — slow
12: active — passive

Table C-5: Individual SD Scale Scores for the Typewriter

S	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1	2	1	1	2	1	1	3	1	2	2	1	1	1	2	2	2	1	3	2
2	3	4	4	3	3	3	1	2	4	2	4	4	1	1	4	4	1	2	2	2
3	1	1	1	1	4	1	1	4	4	1	4	1	1	1	1	1	1	1	3	1
4	2	2	3	1	3	1	1	4	2	3	3	2	2	1	3	3	1	1	4	3
5	2	3	2	2	3	2	2	3	2	2	3	2	2	2	3	3	2	2	2	3
6	1	2	1	1	1	1	1	3	1	2	2	1	1	1	2	2	1	1	4	2
7	2	4	3	1	3	1	2	2	3	2	3	1	2	2	3	3	2	1	3	3
8	1	2	1	1	1	1	1	2	1	1	2	1	1	1	2	2	1	1	3	2
9	2	2	2	1	2	1	1	3	1	3	3	2	1	1	3	4	1	1	4	3
10	3	4	3	1	2	3	1	2	2	3	4	2	2	2	3	4	1	2	4	3
11	3	3	3	3	3	3	1	2	2	3	3	2	2	2	3	3	2	2	3	3
12	4	1	2	1	2	1	1	4	1	3	2	1	2	1	3	3	1	2	4	2
13	2	2	3	3	3	2	2	2	2	3	3	2	2	2	2	4	2	2	4	4
14	2	2	2	1	2	1	1	3	2	2	2	2	1	2	2	2	1	1	3	2
15	3	2	2	1	2	1	1	4	1	1	2	2	1	1	3	2	1	1	4	3
16	2	3	2	1	3	1	1	3	1	2	2	1	1	1	3	3	1	1	3	3
17	2	3	2	2	2	1	2	3	2	3	3	2	2	2	3	3	1	1	3	2
18	2	3	2	1	2	1	1	2	1	3	3	1	1	1	2	3	1	1	4	2
19	3	2	2	1	2	1	1	4	4	3	3	3	3	1	3	3	4	1	2	2
20	1	2	1	1	1	1	1	2	1	1	1	1	1	2	1	1	1	1	2	2
21	3	4	1	1	3	1	1	1	2	3	3	2	1	1	3	3	2	1	4	2

1: I feel calm
2: I feel secure
3: I am tense
4: I am regretful
5: I feel at ease
6: I feel upset
7: I am presently worrying over possible misfortunes
8: I feel rested
9: I feel anxious
10: I feel comfortable
11: I feel self-confident
12: I feel nervous
13: I am jittery
14: I feel "high strung"
15: I am relaxed
16: I feel content
17: I am worried
18: I feel over-excited and "rattled"
19: I feel joyful
20: I feel pleasant

Table C-6: Individual STAI Scale Scores for Etude

S	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1	2	1	1	2	1	1	3	1	2	2	1	1	1	2	2	2	1	3	2
2	4	2	4	1	4	2	1	4	4	3	1	4	3	3	4	2	1	3	4	2
3	1	1	1	1	1	1	1	2	1	2	1	1	1	1	1	2	1	1	3	2
4	3	2	2	1	3	1	1	3	2	3	2	3	2	1	3	3	1	1	4	3
5	3	3	2	1	3	2	2	3	2	2	2	2	2	2	2	2	2	2	3	2
6	3	3	2	1	3	1	1	3	2	3	2	2	2	1	3	4	1	1	4	3
7	1	1	4	1	1	1	1	2	4	1	2	4	4	4	2	3	3	1	3	2
8	4	4	4	3	4	2	1	3	3	2	4	4	4	4	4	4	1	4	2	3
9	4	2	1	1	3	1	1	3	1	2	2	1	1	1	3	3	1	1	4	3
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	2	2
11	2	1	3	2	2	2	1	3	2	2	2	2	2	2	3	2	1	2	3	2
12	3	2	2	1	3	1	1	4	1	3	2	1	2	2	3	2	1	1	3	3
13	3	3	3	2	2	2	2	3	2	3	2	3	3	3	3	3	2	1	4	4
14	3	2	2	1	1	1	1	2	1	2	2	2	3	2	2	3	2	2	4	2
15	4	2	2	1	3	1	1	3	2	2	2	3	2	1	3	3	1	2	4	4
16	3	3	2	1	3	4	1	2	2	3	2	2	1	1	2	3	1	1	4	3
17	2	2	2	1	2	2	2	3	2	3	2	2	2	2	3	3	2	1	3	3
18	2	1	2	1	2	1	1	1	3	1	1	1	1	1	2	2	1	1	4	3
19	1	2	3	1	2	4	1	4	2	2	2	2	1	1	2	2	3	1	4	3
20	3	1	3	1	2	1	1	2	1	3	2	1	2	2	2	2	1	1	2	2
21	1	2	2	1	2	1	2	1	2	1	1	2	1	1	1	2	2	1	4	1

1: I feel calm

2: I feel secure

3: I am tense

4: I am regretful

5: I feel at ease

6: I feel upset

7: I am presently worrying over possible misfortunes

8: I feel rested

9: I feel anxious

10: I feel comfortable

11: I feel self-confident

12: I feel nervous

13: I am jittery

14: I feel "high strung"

15: I am relaxed

16: I feel content

17: I am worried

18: I feel over-excited and "rattled"

19: I feel joyful

20: I feel pleasant

Table C-7: Individual STAI Scale Scores for the Typewriter

S	Tutorial	Typing Task	Editing Task
1	46:25	20:25	9:45
2	1:51:50	23:40	15:30
3	1:31:05	21:45	4:10
4	1:05:45	24:15	8:15
5	1:08:45	18:05	24:45
6	1:16:25	7:20	7:40
7	1:23:55	22:30	14:10
8	1:06:25	32:35	11:45
9	38:20	32:25	14:20
10	25:05	38:25	20:00
11	2:58:00	35:55	7:35
12	49:30	34:00	8:00
13	1:18:25	41:55	15:05
14	51:30	17:40	7:30
15	1:19:50	33:15	8:40
16	1:01:40	16:00	7:45
17	1:00:05	14:35	39:35
18	49:20	23:35	6:00
19	1:35:15	33:20	8:15
20	34:35	23:35	8:50
21	1:03:25	48:05	7:10

Table C-8: Detailed Training Times

Annotated Bibliography

1. Al-Awar, Janan, Chapanis, Alphonse and Ford, W. Randolph. Tutorials for the First-Time Computer User. *IEEE Transactions on Professional Communication PC-24* (1981), 30–37.

Advocates an iterative process for designing tutorials, emphasizing that user mistakes are caused by problems in the tutorial.

2. Baker, James D. and Goldstein, Ira. Batch vs. Sequential Displays: Effects on Human Problem Solving. *Human Factors 8* (1966), 225–235.

Reports an experiment which indicates that the display of currently irrelevant menu items degrades performance.

3. Bennett, John L. The User Interface in Interactive Systems. In *Annual Review of Information Science and Technology, Vol. 7,* C. A. Cuadra, Ed., American Society for Information Science, Washington, 1972, pp. 159–196.

The best survey article for work on user interface design through 1971.

4. Bennett, John L. The Commercial Impact of Usability in Interactive Systems. In *Man/Computer Communication, Vol. 2,* Infotech State of the Art Report, Maidenhead, England, 1979, pp. 1–17.

Discusses the importance of emphasizing usability in commercial development of computer systems, and suggests ways of bringing usability to greater prominence in the development process. One of the best papers in this State of the Art Report.

5. Berman, Lorraine and Karr, Rosemary. Evaluating the "Friendliness" of a Timesharing System. *SIGSOC Bulletin 12* (July 1980), 8–11.

Presents a set of guidelines for friendly systems, based on a study of users working with various timesharing systems.

6. Beyer, William H. (Ed.). *The CRC Handbook of Tables for Probability and Statistics.* The Chemical Rubber Co., Cleveland, 1966.

Table VII.3 on p. 266 gives confidence intervals for medians. Reprinted from K. R. Nair's article.

7. Bobrow, Daniel G. et al. TENEX, a Paged Time Sharing System for the PDP-10. *Communications of the ACM 15* (1972), 135–143.

Origin of the types of prompts and completion found in Etude.

8. Boden, Margaret A. Social Implications of Intelligent Machines. *The Radio and Electronic Engineer 47* (1977), 393–399.

One part of the discussion deals with the social problems of systems that use tricks such as "natural language" to make the system appear more intelligent than it really is.

9. Bott, Ross A. A Study of Complex Learning: Theory and Methodology. Report 82, University of California, San Diego Center for Human Information Processing, March, 1979.

One chapter shows protocols that were recorded when novices were learning the print command in the UNIX editor. Problems include misunderstanding some common computer terms and using close but not quite accurate analogies.

10. Breiman, Leo. *Statistics With a View Toward Applications.* Houghton Mifflin, Boston, 1973.

Pp. 228–239 discuss the use of medians and percentiles; pp. 260–268 discuss the Wilcoxon test.

11. Buros, Oscar Krisen (Ed.). *The Eighth Mental Measurements Yearbook, Vol. I.* The Gryphon Press, Highland Park, N.J., 1978.

Entry No. 683, pp. 1088–1096, includes reviews of the STAI by Ralph Mason Dreger and Edward S. Katkin as well as a bibliography with over 250 entries.

12. Cakir, A., Hart, D. J. and Stewart, T. F. M. *Visual Display Terminals.* John Wiley & Sons, New York, 1980. Originally published in 1979 by The Inca-Fiej Research Association, Darmstadt.

The best reference to date for human factors guidelines in the design of terminals and workstations.

13. Card, Stuart K., English, William K. and Burr, Betty J. Evaluation of Mouse, Rate-Controlled Isometric Joystick, Step Keys, and Text Keys for Text Selection on a CRT. *Ergonomics 21* (1978), 601–613.

Out of these four devices, the mouse is the fastest and most accurate. The experimenters claim that the mouse approaches optimum performance for a pointing device.

14. Card, Stuart K., Moran, Thomas P. and Newell, Allen. The Keystroke-Level Model for User Performance Time with Interactive Systems. *Communications of the ACM 23* (1980), 396–410.

Proposes a model for predicting the time it takes an expert user to perform an error-free task on a system, given the command sequence used. Based primarily on number of keystrokes, but also includes parameters for mental operations.

15. Chamberlin, Donald D. et al. JANUS: An Interactive System for Document Composition. *SIGPLAN Notices 16* (June 1981), 82–91.

JANUS is being developed by a research team at IBM's San Jose facility. Since its functionality overlaps Etude's, it is interesting to compare approaches to many of the same problems.

16. Chapanis, Alphonse. "Words, Words, Words". *Human Factors 7* (1965), 1–17.

Cites many examples of bad instructions and the need for human factors research in this area.

17. *Consistent System: Elementary Statistical Analysis.* First edition, Renaissance Computing, Inc., 675 Massachusetts Avenue, Cambridge, MA 02139, 1980.

18. Cuff, Rodney N. On Casual Users. *International Journal of Man-Machine Studies 12* (1980), 163–187.

Analyzes the characteristics and needs of casual or computer-naive users, and uses this to motivate some guidelines for user interface design, especially in the area of query languages.

19. Davies, Donald W. and Yates, David M. Human Factors in Display Terminal Procedures. Proc. Fourth International Conference on Computer Communication, International Council for Computer Communication, September, 1978, pp. 777–783.

Discusses question and answer, menu, and form-filling dialogue techniques as well as style of interaction.

20. Deese, James. The Associative Structure of Some Common English Adjectives. *Journal of Verbal Learning and Verbal Behavior 3* (1964), 347–357.

Includes a list of bipolar adjectives, derived from word association experiments. Useful in connection with building an SD.

21. DiVesta, Francis J. A Developmental Study of the Semantic Structures of Children. *Journal of Verbal Learning and Verbal Behavior 5* (1966), 249–259.

One of the best sources of SD scales for children. The study referenced in this thesis used second through seventh graders as subjects, evaluating 100 concepts with 20 scales.

22. Dzida, W., Herda, S. and Itzfeldt, W. D. User-Perceived Quality of Interactive Systems. *IEEE Transactions on Software Engineering SE-4* (1978), 270–276.

Describes seven factors of user-perceived quality, five of which have internal validity and consistency. These factors were gathered from a questionnaire sent to hundreds of German computer users.

23. Ehardt, Joseph L. and Seybold, Patricia B. Experimental Systems: Xerox Document System, M.I.T. Etude. *The Seybold Report on Word Processing 3*, 9 (October 1980).

A description of Etude from a perspective other than that of the designers.

24. Embley, David W. and Nagy, George. Behavioral Aspects of Text Editors. *ACM Computing Surveys 13* (1981), 33–70.

A comprehensive survey of work done in the area, with 121 references.

25. Engel, Stephen E. and Granda, Richard E. Guidelines for Man/Display Interfaces. Tech. Rep. TR 00.2720, IBM Poughkeepsie Laboratory, December 19, 1975.

Gives a set of guidelines in the areas of display frame layout, frame content, command languages, error prevention and recovery, response time, and behavioral principles. One of the most frequently cited sources of such guidelines.

26. Fields, Allson F., Malsano, Richard E. and Marshall, Charles F. A Comparative Analysis of Methods for Tactical Data Inputting. Technical Paper 327, U. S. Army Research Institute for the Behavioral and Social Sciences, Sept, 1978. NTIS No. AD-A060 562.

Compares plain typing, typing with spelling correction, typing with completion and optional English (rather than codes), and menus. Menus are more accurate, while no significant differences in time were measured among the four. Lack of time difference may be due to lack of time (one day) using the various techniques. Menus used trackballs for selection, a choice that is criticized in the discussion section.

27. Foley, James D. and Wallace, Victor L. The Art of Natural Graphic Man-Machine Conversation. *Proceedings of the IEEE 62* (1974), 462–471.

Illustrates psychological blocks of boredom, panic, frustration, confusion, and discomfort. Proposes a structure for action languages, including virtual input devices, that has evolved into the Core Graphics Standard.

28. Frierson, Elanor and Atherton, Pauline. Survey of Attitudes Towards SUPARS. Proceedings of the American Society for Information Science, Vol. 8, Greenwood Publishing Co., Westport, Conn., 1971, pp. 65–69.

An early study using the SD to evaluate a computer system.

29. Gaines, Brian R. The Technology of Interaction—Dialogue Programming Rules. *International Journal of Man-Machine Studies 14* (1981), 133–150.

Expansion and revision of rules proposed in Gaines and Facey's 1975 paper, including six more guidelines. This should be the starting point for someone new to the study of user interface design.

30. Gaines, Brian R. and Facey, Peter V. Some Experience in Interactive System Development and Application. *Proceedings of the IEEE 63* (1975), 894–911.

The last part of this paper presents eleven guidelines for designing interactive systems. Some authors have used this paper as the basis for papers of their own. Cited very frequently.

31. Galley, S. W. and Pfister, Greg. *The MDL Programming Language.* MIT Lab. for Computer Science, 1979.

32. Gebhardt, Friedrich and Stellmacher, Imant. Design Criteria for Documentation Retrieval Languages. *Journal of the American Society for Information Science 29* (1978), 191–199.

Examines the tradeoffs between various design criteria for easy to use systems. Concludes that the tradeoff between simplicity for the casual user and flexibility for the experienced user is the most severe problem.

33. Gibson, R. An Annotated Bibliography of Man/Computer Communication. In *Man/Computer Communication, Vol. 1,* Infotech State of the Art Report, Maidenhead, England, 1979, pp. 301–337.

A good bibliography with a European accent. 78 of the 204 entries are annotated.

34. Gilb, Tom and Weinberg, Gerald M. *Humanized Input.* Winthrop, Cambridge, Mass., 1977.

Subtitled "Techniques for Reliable Keyed Input", contains a wealth of information and recommendations for data entry design. Emphasizes the importance of reliability.

35. Goldfarb, C. S. A Generalized Approach to Document Markup. *SIGPLAN Notices 16* (June 1981), 68–73.

Describes GML, another formatter that works with editorial structure, though not as well as Scribe.

36. Good, Michael. A Programmer's Guide to Etude. Memo OAM-014, MIT Lab. for Computer Science, Office Automation Group, April, 1980.

37. Good, Michael. Etude and the Folklore of User Interface Design. *SIGPLAN Notices 16* (June 1981), 34–43.

Discusses in detail how some of Etude's major features meet user interface design guidelines. Includes quotations from the literature.

38. Goodman, T. J. and Spence, Robert. The Effect of Computer System Response Time Variability on Interactive Graphical Problem Solving. *IEEE Transactions on Systems, Man, and Cybernetics SMC-11* (1981), 207–216.

Response time variability did not have a significant effect on task completion time in this experiment. Authors suggest that the effect of response time variability may be task dependent.

39. Hammer, Michael et al. The Implementation of Etude, An Integrated and Interactive Document Production System. *SIGPLAN Notices 16* (June 1981), 137–146.

Describes the implementation of the first version of Etude.

40. Hammer, Michael et al. Etude: An Integrated Document Processing System. 1981 Office Automation Conference Digest, AFIPS, March, 1981, pp. 209–219.

Explains the basic ideas of Etude.

41. Hansen, Joe B. Effects of Feedback, Learner Control, and Cognitive Abilities on State Anxiety and Performance in a Computer-Assisted Instruction Task. *Journal of Educational Psychology 66* (1974), 247–254.

Reports an experiment where feedback in CAI reduced state anxiety as measured by the STAI.

42. Hansen, Wilfred J. User Engineering Principles for Interactive Systems. AFIPS Conference Proceedings, Vol. 39, AFIPS Press, Montvale, N. J., 1971, pp. 523–532.

One of the first papers to describe a structured editor and to present user interface guidelines. Emphasizes the need to know the user of a system.

43. Hayes, Phil, Ball, Eugene, and Reddy, Raj. Breaking the Man-Machine Communication Barrier. *Computer 14* (March 1981), 19–30.

Gives a fine example of the communication problems in current computer systems, then proposes ways around these problems. Includes a summary of what the authors term "graceful interaction."

44. Hebditch, David. Design of Dialogues for Interactive Commercial Applications. In *Man/Computer Communication, Vol. 2,* Infotech State of the Art Report, Maidenhead, England, 1979, pp. 171–192.

Discusses dialogue design methodology and techniques. Interesting in that many of his opinions run contrary to that of several other authors.

45. Heise, David R. Some Methodological Issues in Semantic Differential Research. *Pschological Bulletin 72* (1969), 406–422.

The second best paper that discusses methodological issues conerning the SD. For

the best paper, see the next entry.

46. Heise, David R. The Semantic Differential and Attitude Research. In *Attitude Measurement,* Gene F. Summers, Ed., Rand McNally, Chicago, 1970, pp. 235–253.

A must for anyone doing SD research. Details methodological considerations in constructing and administering SD's, citing many different studies. Can be used as a checklist of things to consider when using an SD in an experiment.

47. Hill, I. D. Wouldn't It Be Nice If We Could Write Computer Programs in Ordinary English—Or Would It? *The Computer Bulletin 16* (1972), 306–312.

An informal article that illustrates the ambiguities of using English for specifying instructions.

48. Ilson, Richard. An Integrated Approach to Formatted Document Production. Tech. Rep. TR-253, MIT Lab. for Computer Science, August, 1980. MIT M.S. thesis.

General description of Etude, including derivations from earlier systems and considerable implementation detail.

49. Ilson, Richard and Good, Michael. Etude: An Interactive Editor and Formatter. Memo OAM-029, MIT Lab. for Computer Science, Office Automation Group, March, 1981. Revised May 1981.

The current Etude specifications.

50. Jackobovits, Leon A. Comparative Psycholinguistics in the Study of Cultures. *International Journal of Psychology 1* (1966), 15–37.

Probably the best source of SD scales, due to the extensive cross-cultural research that supports it. Fifteen scales are presented for each of fifteen language-culture communities. Scales are drawn from experiments involving 50 scales × 100 concepts × 20 subjects.

51. Jones, P. F. Four Principles of Man-Computer Dialogue. *Computer Aided Design 10* (1978), 197–202.

The four principles are expectation, implication, experimentation, and motivation.

52. Kendall, Maurice G. *Rank Correlation Methods.* Hafner Publishing Co., New York, 1962. Third edition.

Chapter 6, "The Problem of *m* Rankings," describes *W*, Kendall's coefficient of concordance, used in construction of an SD.

53. Kennedy, T. C. S. The Design of Interactive Procedures for Man-Machine Communication. *International Journal of Man-Machine Studies 6* (1974), 309–334.

Proposes a set of twelve ground rules for the design of a "well-behaved" system.

54. Kennedy, T. C. S. Some Behavioural Factors Affecting the Training of Naive Users of an Interactive Computer System. *International Journal of Man-Machine Studies 7* (1975), 817–834.

Recounts the success of a training system for hospital workers.

55. Kinkead, Robin. Typing Speed, Keying Rates, and Optimal Keyboard Layouts. Proceedings of the Human Factors Society 19th Annual Meeting, October, 1975, pp. 159–161.

Describes experiments that indicate that the standard keyboard is operated at near-maximum rates. Claims that larger speed increases can be made by eliminating keystrokes than by redesigning the keyboard.

56. Ledgard, Henry et al. The Natural Language of Interactive Systems. *Communications of the ACM 23* (October 1980), 556–563.

A line editor with an English-like syntax outperformed an editor with the same functionality but a more conventional "notational" syntax. These gains held for users of all experience levels.

57. Ledgard, Henry, Singer, Andrew and Whiteside, John. *Directions in Human Factors for Interactive Systems.* Springer-Verlag, New York, 1981.

Valuable especially for an annotated version of the experimental diary that recorded the progress of the experiment reported above. Has other good sections on experimental topics.

58. Levitt, Eugene E. *The Psychology of Anxiety.* Lawrence Erlbaum Associates, Hillsdale, N. J., 1980. Second Edition.

A readable introduction to anxiety theory and research. Chapter 5 on the experimental measurement of anxiety is the most directly useful part of the book. Large bibliography.

59. Licklider, J. C. R. Man-Computer Symbiosis. *IRE Transactions on Human Factors in Electronics HFE-1* (1960), 4–11.

The first paper to deal with the ease of use of computer systems, its inclusion in a bibliography such as this is practically mandatory.

60. Liskov, Barbara, et al. *CLU Reference Manual.* Springer-Verlag, New York, 1981.

61. Lucas, R. W. A Study of Patients' Attitudes to Computer Interrogation. *International Journal of Man-Machine Studies 9* (1977), 69–86.

Should be read by anyone measuring attitudes towards computer systems. Shows how two different types of attitude scales (a traditional Thurstone scale and an SD) are constructed.

62. Malone, Thomas W. What Makes Things Fun to Learn? A Study of Intrinsically Motivating Computer Games. Report CIS-7 (SSL-80-11), Xerox Palo Alto Research Center, August, 1980. Slightly revised version of Stanford Ph.D. dissertation.

Examines how the elements of challenge, fantasy, and curiosity make (computer) games intrinsically motivating. Stresses the importance of goals in games, and suggests ways to make teaching of specific skills more interesting.

63. Mann, William C. Why Things Are So Bad for the Computer-Naive User. Tech. Rep. ISI/RR-75-32, University of Southern California Information Sciences Institute, March, 1975.

Why? Because all that most interfaces use are commands, while people state goals, give examples, describe, clarify, hypothesize, use analogies, make comparisons, etc., in addition to using commands a small part of the time. Provocative reading.

64. Margulies, F. Technological Change: Its Impact on Man and Society. In *Man/ Computer Communication, Vol. 2,* Infotech State of the Art Report, Maidenhead, England, 1979, pp. 251–261.

Emphasizes the importance of evaluation criteria that reflect the impact of systems on the people that use them and on society at large.

65. Martin, James. *Design of Man-Computer Dialogues.* Prentice-Hall, Englewood Cliffs, N.J., 1973.

One of the earliest books on the topic, discussing many issues dealing with techniques of dialogue design. Chapter 7 on techniques for alphanumeric keyboard displays and Chapter 8 on control functions are of special interest.

66. McCormick, Ernest J. *Human Factors in Engineering and Design.* McGraw-Hill, New York, 1976. Fourth Edition.

The most recent edition of a well known human factors text. Chapters 3 and 4 (on information processing), and 10 and 11 (on work space and arrangement) are the most applicable for computer system designers.

67. Miller, Lance A. and Thomas, John C., Jr. Behavioral Issues in the Use of Interactive Systems. *International Journal of Man-Machine Studies 9* (1977), 509–536.

Concise overview of the field, with 142 references.

68. Miller, Lawrence H. A Study in Man-Machine Interaction. AFIPS Conference Proceedings, Vol. 46, AFIPS Press, Montvale, N.J., May, 1977, pp. 409–421.

Presents experimental evidence that variability of computer output both degrades performance and results in poorer user attitudes. Doubling the display rate from 1200 to 2400 baud had no significant effect in either area.

69. Miller, Lawrence H. A Resource for Investigating Human Interaction with Computers. In *Teleinformatics '79,* E. J. Boutmy and A. Danthine, Eds., North-Holland, Amsterdam, 1979, pp. 195–200.

Includes a discussion of the utility of eight different peformance and productivity measures.

70. Miller, Robert B. Response Time in Man-Computer Conversational Transactions. AFIPS Conference Proceedings, Vol. 33, Part 1, Thompson Book Co., Washington, 1968, pp. 267–277.

Gives suggested bounds for response times needed by a user in different situations. Anything over 15 seconds can destroy the conversational nature of the system, so the user should be freed to do something else if that speed cannot be met in a certain instance. A classic in the literature of response time.

71. Miller, Robert B. Human Ease of Use Criteria and Their Tradeoffs. Tech. Rep. TR 00.2185, IBM Poughkeepsie Laboratory, April 12, 1971.

Still among the finest papers which examine the question of what is meant by ease of use. Both of Bennett's papers present modified versions of this work and are more readily available, but the original is worth seeking out.

72. Mitsos, Spiro B. Personal Constructs and the Semantic Differential. *Journal of Abnormal and Social Psychology 62* (1961), 433–434.

Provides experimental evidence of the importance of relevance of SD scales to the concepts being measured.

73. Nair, K. R. Table of Confidence Interval for the Median in Samples from Any Continuous Population. *Sankhya, The Indian Journal of Statistics 4* (1940), 551–558.

74. Neale, John M. and Liebert, Robert M. *Science and Behavior.* Prentice-Hall, Englewood Cliffs, N.J., 1973.

Discusses several major aspects of experimental research. Chapter 3 deals with internal validity, Chapter 9 with external validity.

75. Newman, William M. and Sproull, Robert F. *Principles of Interactive Computer Graphics.* McGraw-Hill, New York, 1979. Second Edition.

One of the basic books in the area of computer graphics. For the user interface designer, chapter 28 is of primary interest.

76. Niamir, Bahram. A Virtual Terminal Interface for Text Processing Applications. Memo OAM-011, MIT Lab. for Computer Science, Office Automation Group, December, 1979.

77. Nickerson, Raymond S. and Pew, Richard W. Oblique Steps Toward the Human-Factors Engineering of Interactive Computer Systems. Appendix to Bolt Beranek and Newman Report No. 2190 by Mario C. Grignetti et al., *Information Processing Models and Computer Aids for Human Performance,* June 30, 1971. NTIS No. AD 732 913.

Another useful potpourri of ideas about the user interface. Makes several references to TENEX.

78. Office Automation Group. Annual Progress Report. Memo OAM-017, MIT Lab. for Computer Science, Office Automation Group, June, 1980.

79. Osgood, Charles E. Studies on the Generality of Affective Meaning Systems. *American Psychologist 17* (1962), 10–28.

More studies from the father of the SD.

80. Osgood, Charles E., Suci, George J. and Tannenbaum, Percy H. *The Measurement of Meaning.* University of Illinois Press, 1957.

The basic book on the SD. Required reading for anyone using this technique.

81. Palme, Jacob. Interactive Software for Humans. *Management Datamatics 5,* 4 (1976), 139–154.

Section 5 on interactive techniques contains some useful ideas.

82. Pew, Richard W. and Rollins, Ann M. Dialog Specification Procedures. Report 3129, Bolt Beranek and Newman, September, 1975. Revised ed. NTIS No. PB-252 976.

Describes general principles of dialogue design, then gives recommendations for the specific system being developed.

83. Plum, Thomas. Fooling the User of a Programming Language. *Software—Practice and Experience 7* (1977), 215–222.

Describes problems with "natural" constructs. Solution—either a simple language devoid of user-fooling powers, or an explication language for more complex languages.

84. Ramsey, H. Rudy and Atwood, Michael E. Human Factors in Computer Systems: A Review of the Literature. Tech. Rep. SAI-79-111-DEN, Science Applications, Inc., September, 1979. NTIS No. AD-A075 679.

Comments on the availability and quality of literature in many different areas in the design of systems. Summarizes the best that is available from the next entry.

85. Ramsey, H. Rudy, Atwood, Michael E. and Kirshbaum, Priscilla J. A Critically Annotated Bibliography of the Literature on Human Factors in Computer Systems. Tech. Rep. SAI-78-070-DEN, Science Applications, Inc., May, 1978. NTIS No. AD-A058 081.

A very useful bibliography with over 500 entries, indexed by author and subject. But beware—if you have a microfiche copy, the small, upper case only letters make it very hard to read. A peculiar fault for a human factors bibliography.

86. Reid, Brian K. A High-Level Approach to Computer Document Formatting. Conference Record of the Seventh Annual ACM Symposium on Principles of Programming Languages, ACM, January, 1980, pp. 24–31.

Introduces the concept of high-level formatting, as done in Scribe.

87. Reid, Brian K. and Hanson, David. An Annotated Bibliography of Background Material on Text Manipulation. *SIGPLAN Notices 16* (June 1981), 157–160.

A selected bibliography including material on typography, writing style, etc.

88. Reisner, Phyllis. Uses of Psychological Experimentation as an Aid to Development of a Query Language. *IEEE Transactions on Software Engineering SE-3* (1977), 218–229.

Describes paper and pencil experiments used in evaluating the SEQUEL query language for purposes such as identifying error-prone constructs.

89. Relles, Nathan and Price, Lynne A. A User Interface for Online Assistance. Proceedings of the Fifth International Conference on Software Engineering, IEEE Computer Society Press, March, 1981, pp. 400–408.

This version of help provides several different kinds of help at the user's request. Help files are kept in scripts. Contains suggestions for the wording of help messages.

90. Roberts, Teresa L. Evaluation of Computer Text Editors. Report SSL-79-9, Xerox Palo Alto Research Center, November, 1979. Stanford Ph.D. dissertation.

Proposes a mechanism for comparing text editors. Editors evaluated are TECO, Wylbur, NLS, and Wang. The use of four subjects makes the mechanism quick to use but limits its power. Includes a standard teaching method and a functional checklist.

91. Robertson, G., McCracken, D. and Newell, A. The ZOG Approach to Man-Machine Communication. *International Journal of Man-Machine Studies 14* (1981), 461–488.

Describes the menu-driven ZOG system and attempts to make this a model of man-computer communication.

92. Rohlfs, S. User Interface Requirements. In *Convergence, Vol. 2,* Infotech State of the Art Report, Maidenhead, England, 1979, pp. 165–199.

Discusses many different aspects of the user interface, including workstation design, dialogue issues, and user participation in the design process. A different point of view than provided in most sources of guidelines. Valuable reading.

93. Rohlfs, Sabine. Linguistic Considerations for User Interface Design. Integrated Office Systems—Burotics, North-Holland, Amsterdam, 1980, pp. 189–196. Proceedings of the IFIP TC-6 Workshop, Versailles, November 6–9, 1979.

Illustrates problems when word meanings differ between natural language and a user language. Suggests ways to understand and explain these differences.

94. Rouse, William B. Design of Man-Computer Interfaces for On-Line Interactive Systems. *Proceedings of the IEEE 63* (1975), 847–857.

Contains general recommendations for interfaces as well as some specific ones for input and display devices. Large bibliography in the area of visual information processing.

95. Runyon, Richard P. *Nonparametric Statistics: A Contemporary Approach.* Addison-Wesley, 1977.

Table J, critical values for *T* in the Wilcoxon test, is more conservative than Wilcoxon and Wilcox's table.

96. Seybold, Jonathan. The Xerox 'Professional Workstation'. *The Seybold Report 10,* 16 (April 27, 1981).

Describes the Xerox Star. A slightly different version of this article appears in the May 1981 issue of the *Seybold Report on Word Processing.*

97. Shackel, B. Dialogues and Language—Can Computer Ergonomics Help? *Ergonomics 23* (1980), 857–880.

Surveys how work in human factors can be applied to dialogue design.

98. Shneiderman, Ben. *Software Psychology.* Winthrop, Cambridge, Mass., 1980.

Subtitled *Human Factors in Computer and Information Systems,* this book emphasizes experimental results in human factors research. Although primarily oriented towards programming, Chapters 10, 11, and 12 are more generally concerned with interactive systems.

99. Shneiderman, Ben. Human Factors Studies with System Message Styles. Talk presented at Joint Conference on Easier and More Productive Use of Computing Systems, May 20–22, 1981, Ann Arbor, Mich.

Improved specificity in system messages improves both user performance and satisfaction. Changing the tone of messages has a less clear effect.

100. Slobin, Dan Isaac. *Psycholinguistics.* Scott, Foresman and Co., Glenview, Ill., 1979. Second edition.

101. Smith, Hugh. Human-Computer Communication. In *Human Interaction with Computers,* H. T. Smith and T. R. G. Green, Eds., Academic Press, New York, 1980, pp. 5–38.

An introduction to the collection and thus a summary article, emphasizing social issues and (un)natural language interfaces.

102. Smith, Hugh T. and Green, T. R. G. (Eds.). *Human-Computer Interaction.* Academic Press, New York, 1980.

A collection of papers by various authors on topics such as social issues, computer applications, and improving programmer performance.

103. Snider, James G. and Osgood, Charles E. (Eds.). *Semantic Differential Technique.* Aldine Publishing Co., Chicago, 1969.

Reprints over fifty articles about the SD (including papers by Deese and DiVesta), and includes a large bibliography. A valuable reference.

104. Spielberger, Charles D. Anxiety as an Emotional State. In *Anxiety: Current Trends in Theory and Research, Vol. 1,* C. D. Spielberger, Ed., Academic Press, New York, 1972, pp. 23–49.

Describes state anxiety (A-State), trait anxiety (A-Trait), and the development of the STAI. Includes several references to evidence of the construct validity of the STAI A-State scale.

105. Spielberger, Charles D., Gorsuch, Richard L. and Lushene, Robert E. *Manual for the State-Trait Anxiety Inventory.* Consulting Psychologists Press, 577 College Ave., Palo Alto, Calif. 94306, 1970.

Test manual for form X. Available only from publisher.

106. Spielberger, Charles D., O'Neil, Harold F., Jr. and Hansen, Duncan N. Anxiety, Drive Theory, and Computer-Assisted Learning. In *Progress in Experimental Personality Research, Vol. 6,* Brendan A. Maher, Ed., Academic Press, New York, 1972, pp. 109–148.

Presents the results of four experiments in this area. These generally support the theory that subjects with low state anxiety will perform better in learning experiments than those with high state anxiety on difficult tasks, but that the results may be reversed on easy tasks.

107. Spiliotopoulos, V. and Shackel, B. Towards a Computer Interview Acceptable to the Naive User. *International Journal of Man-Machine Studies 14* (1981), 77–90.

A recent study using the SD to judge attitudes toward computer interviews.

108. Stallman, Richard M. EMACS, the Extensible, Customizable Self-Documenting Display Editor. *SIGPLAN Notices 16* (June 1981), 147–156.

109. Stewart, Tom. Communicating with Dialogues. *Ergonomics 23* (1980), 909–919.

Gives ten guidelines for dialogue design, the most novel of which is not to depend solely on general principles and guidelines.

110. Teitelman, Warren. INTERLISP *Reference Manual.* Xerox Palo Alto Research Center, December 1975. Second revision.

Describes a very thorough undo and history facility in sections 9 and 22.

111. Thimbleby, Harold. Dialogue Determination. *International Journal of Man-Machine Studies 13* (1980), 295–304.

Introduces principle of determination. Restricted syntax tends to over-determine; obscure or highly flexible systems tend to under-determine. A system may be both over and under-determined, making it ill-determined in general. Discusses effects of flexibility, commonality, immediacy, intromission, variability, and feedback.

112. Thomas, John C., Jr. A Design-Interpretation Analysis of Natural English with Applications to Man-Computer Interaction. *International Journal of Man-Machine Studies 10* (1978), 651–668.

Argues that human communication is designed and interpreted according to goals and other criteria, rather than simply encoded and decoded as other authors suggest.

This has several effects on man-computer interaction, including the desirability of a well-designed metacomment facility.

113. Treu, Sigfried. Interactive Command Language Design Based on Required Mental Work. *International Journal of Man-Machine Studies 7* (1975), 135–149.

Describes action primitives, a sequence of 1) action verb, 2) action qualifier(s), 3) object(s) of the action, and 4) object qualifier(s). Hypothesizes that the closer the set of system commands is to the set of action primitives, the less mental work is required to use the command language.

114. Uttal, Bro. The Coming Struggle in Personal Computers. *Fortune 13* (June 29, 1981), 84–92.

115. Walther, George H. The On-Line User-Computer Interface: The Effects of Interface Flexibility, Experience, and Terminal-Type on User-Satisfaction and Performance. Ph.D. Th., University of Texas at Austin, August, 1973. NTIS No. AD-777 314.

Valuable for several reasons. This was one of the first studies to use questionnaires such as the STAI and the SD in addition to speed measurements. Also shows the pitfalls that await the experimenter who tries to do too much at once—the data analysis is rather jumbled.

116. Walther, George H. and O'Neil, Harold F., Jr. On-Line User-Computer Interface—The Effects of Interface Flexibility, Terminal Type, and Experience on Performance. AFIPS Conference Proceedings, Vol. 44, AFIPS Press, Montvale, N.J., May, 1974, pp. 379–384.

Brief summary of Walther's Ph.D. thesis.

117. Ward, Stephen A. and Terman, Christopher J. An Approach to Personal Computing. Digest of Papers, Compcon '80, IEEE, February, 1980, pp. 460–465.

Describes the Nu machine.

118. Wasserman, Anthony I. User Software Engineering and the Design of Interactive Systems. Proceedings of the Fifth International Conference on Software Engineering, IEEE Computer Society Press, March, 1981, pp. 387–393.

Present some design guidelines, then goes on to describe a system that facilitates building of prototype interfaces.

119. Watson, Richard William. User Interface Design Issues for a Large Interactive System. AFIPS Conference Proceedings, Vol. 35, Thompson Book Co., Washington, 1976, pp. 357–364.

Describes the user interface of NLS, both in terms of design goals and specific details. Includes ideas of a common interface among tools and a verb-object syntax.

120. Wilcoxon, Frank. Individual Comparisons by Ranking Methods. *Biometrics Bulletin 1* (1945), 80–83.
Describes the theory behind the Wilcoxon test.

121. Wilcoxon, Frank and Wilcox, Roberta A. *Some Rapid Approximate Statistical Procedures.* American Cyanamid Co., Pearl River, N.Y., 1964.
Describes how to use the Wilcoxon test.

122. Wilde, G. J. S. Trait Description and Measurement by Personality Questionnaires. In *Handbook of Modern Personality Theory,* R. B. Cattell and R. M. Dreger, Eds., Hemisphere Publishing Corp., Washington, 1977, pp. 69–103.
Enumerates and responds to objections to the validity of using questionnaires in measuring personality traits, concluding that there is a firm empirical basis for their use in this area.

Index